Universidad de las Ciencias Informáticas
Facultad 1

Título: “Sistema de evaluación de seguridad utilizando el proceso de integración de datos ETL”

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autores:

Yanisleydis Romero Peña
Jorge Fonseca Martínez

Tutor(es):

Ing. Yayneris Zambrana Hernández
Ing. Geidis Sánchez Michel

La Habana, Cuba
Año 54 de la Revolución

"Seamos realistas y hagamos lo imposible."
Ernesto Che Guevara.

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores del trabajo titulado: Sistema de evaluación de seguridad utilizando el proceso de integración de datos ETL, y autorizamos a la Universidad de las Ciencias Informáticas los derechos patrimoniales del mismo, con carácter exclusivo.

Para que así conste firmamos la presente a los _____ días del mes de _____ del año _____.

_____ _____

Yanisleydis Romero Peña. Jorge Fonseca Martínez.

_____

Ing. Yayneris Zambrana Hernández.

_____

Ing. Geidis Sánchez Michel.

DEDICATORIA

Dedico este trabajo de diploma a dos personas muy importantes que ya no están conmigo, mi tía Flor María y mi papá Juan Rodolfo, dondequiera que se encuentren deben estar muy orgullosos por lo que he logrado ser, los extraño mucho.

A mi mamá por darme tanto apoyo durante estos cinco años en las buenas y en las malas, por aguantar mis malcriadeces y por ser la persona más importante en mi vida.

A Kiko mi papá dos como le llamo cariñosamente, por quererme y cuidarme como si fuera su hija, a mis abuelitos que los adoro con todo mi corazón y a toda mi familia.

YANISLEYDIS

Le dedico este trabajo de diploma y todo lo que soy a las personas más importantes de mi vida, las que me apoyaron y creyeron en mi todo el tiempo, las que me aconsejaban cuando las cosas no me salían bien, las que me decían no te preocupes que de todo se sale. En fin le dedico todo esto por tantas y tantas cosas, pero la principal y más importante porque los quiero mucho.

Realmente creo que cada hijo en este mundo tiene que estarle agradecido a sus padres por cada cosa, quizás haya niños que ya no los tengan, pero yo le doy gracias a Dios porque los tengo a ambos y porque todavía puedo compartir cada minuto de mi vida con ellos; por eso les dedico todo lo que soy, para poder de alguna forma expresarle todo lo que siento por ellos.

Esas personas de quien hablo son mi mamá, y mi papá, los merecedores de todas mis victorias, a ellos va dedicado todo lo que soy. Los quiero mucho.

JORGE

AGRADECIMIENTOS

Agradecer infinitamente a mi mamá y a Kiko por ser mi principal apoyo durante estos años, gracias a ellos llegué tan lejos.

A mis abuelitos, mi tía Yudi por su cariño infinito y a todos los que en mi familia me ayudaron a ser fuerte en los momentos difíciles y formaron parte de esta trayectoria.

A mi novio Yunior por darme todo el amor del mundo y muchas fuerzas para seguir adelante siempre que lo necesité, te quiero mucho.

A Martha, Evelio, Mayelín y Evelito por brindarme y acogerme en su hogar como si fuera parte de la familia.

A mis suegros que también estuvieron pendientes en todo momento de mis resultados y me brindaron mucho cariño.

A los muchachos del apartamento de mi novio por hacer que estos años fueran más divertidos cuando el estrés estaba al máximo.

A mi compañero de tesis Jorge por darme ánimo antes las dificultades y por hacer posible la realización de este trabajo.

A todos mis compañeros de aulas con los que compartí momentos difíciles e inolvidables y que me ofrecieron su ayuda incondicional.

Muchas gracias.

YANISLEYDIS

AGRADECIMIENTOS

Fueron muchas las personas que tuvieron que ver con este gran logro, pero sobresalen mi mamá, mi papá y mi hermana que sacrificaron mucho para que llegara este momento y me dieron su apoyo incondicional en cualquier decisión que tomara.

Le agradezco a mi novia Evelyn por estar ahí siempre para mí, por confiar y aguantar mi carácter y mis malos ratos.

A mi suegra Estrella que es como una madre para mi, a mi cuñado Roniel, a mi tío Antonio por apoyarme mucho en el primer año de la carrera, a mi suegro Carlos, a mi tía Octavia, a mi tío Misael por estar siempre ahí cuando hizo falta y contribuir a que este día llegara. A mi familia en general le agradezco mucho por darme su apoyo cuando lo necesité.

Le agradezco a mi compañera de tesis Yanisleydis que sin ella no habría sido posible todo esto.

Le agradezco a mis compañeros de toda la carrera, a los que se mantienen y los que ya no están, por compartir los mejores años de la juventud, por compartir alegrías, tristezas, buenos y malos momentos, por la ayuda que me dieron cuando hizo falta.

Muchas gracias

JORGE

A las tutoras Yayneris y Geidis por ayudarnos tanto y ser parte de cada momento difícil por el que pasamos, especialmente a Yayneris.

A todo los que de una forma u otra hicieron posible la realización de este trabajo.

AUTORES

RESUMEN

El uso de herramientas para realizar pruebas de seguridad, trae consigo resultados que brindan información referente a las vulnerabilidades de un sistema informático. Muchas veces estos resultados no son procesados debidamente, para obtener una evaluación general de la seguridad de una aplicación. Con el objetivo de que estos datos sean utilizados para dar un criterio sobre el nivel de seguridad de las aplicaciones desarrolladas en el Departamento de Seguridad Digital, perteneciente al Centro de Identificación y Seguridad Digital (CISED), surge el presente trabajo de diploma titulado "Sistema de evaluación de seguridad utilizando el proceso de integración de datos ETL[1]", el cual propone el desarrollo de una aplicación web para facilitar dicha evaluación.

La investigación del trabajo estuvo enmarcada fundamentalmente en el estudio de los estándares que propone la Organización Internacional para la Estandarización (ISO) y las métricas establecidas por estos para el desarrollo de aplicaciones seguras. En el análisis de herramientas de pruebas de seguridad y de los resultados arrojados por estas, los cuales permiten identificar vulnerabilidades. Además, en el estudio de las principales herramientas, tecnologías y metodología para el desarrollo de aplicaciones web, así como en la definición de las características del sistema, planificación, desarrollo y pruebas necesarias para garantizar la calidad del producto final.

PALABRAS CLAVE

Evaluación de seguridad, métricas de seguridad, pruebas, vulnerabilidades.

[1] Por sus siglas en inglés Extracción, Transformación y Carga.

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

En las últimas décadas, con el adelanto de la informatización en todas las esferas de la sociedad y el perfeccionamiento acelerado de las Tecnologías de la Información y las Comunicaciones (TIC), el desarrollo de *software* se ha convertido en un elemento de gran importancia en el mundo. Esto se debe a que la mayoría de las empresas se han visto en la necesidad de informatizar los procesos de trabajo que desarrollan para ganar en competitividad, eficiencia y tiempo.

Para lograr que estos aspectos se cumplan, el proceso de creación de este tipo de aplicaciones informáticas ha de ir acompañado de una actividad que garantice la calidad; teniendo en cuenta que la misma permite conocer las cualidades que deben presentar los productos o servicios, enfocados a satisfacer las necesidades del cliente.

La calidad y seguridad del *software* es un campo que ha tenido una fructífera actividad investigadora en los últimos años, estando principalmente centrada en la calidad de los procesos que se siguen para desarrollar el *software*. Prueba de ello es la gran cantidad de modelos y estándares de referencia, evaluación y mejora de procesos de *software* que han surgido durante las últimas décadas. (1)

La ISO en su estándar ISO/IEC 9126-1 define el término calidad como un conjunto estructurado de características y sub-características, que además pueden ser medidas mediante el uso de métricas. Entre las características que se definen, se encuentra la funcionalidad: “Capacidad del *software* para proporcionar funciones que satisfacen las necesidades declaradas e implícitas cuándo el *software* se usa bajo las condiciones especificadas” (2). Y a su vez dentro de esta característica, se define como sub-características la seguridad: “Capacidad del producto de *software* para proteger información y los datos, para que personas o sistemas desautorizados no puedan leer o pueden modificar los mismos, y las personas o sistemas autorizados tengan el acceso a ellos” (2).

Con el surgimiento de la ISO además de estándares de calidad se han publicado diversos estándares relativos a temas de seguridad, los cuales han surgido con el objetivo de fortalecer los productos de *software*, basándose en la idea de que abordar la seguridad en las fases del ciclo de vida del *software*

tiene un impacto muy significativo en la reducción de vulnerabilidades, y consecuentemente es más eficaz respecto a los costes y tiene como resultado diseños más robustos y sistemas más seguros. (1)

El estudio de estos estándares debe ser un punto relevante en todas aquellas instituciones donde el desarrollo de *software* sea la actividad fundamental. En Cuba actualmente existen diversos centros productores de *software*, ejemplo de ello es la Universidad de las Ciencias informáticas (UCI).

La UCI surge en el 2002 con la misión de formar profesionales altamente calificados, producir *software* y servicios informáticos, a partir de la vinculación estudio-trabajo como modelo de formación. Como institución para la producción de *software* desempeña un papel significativo, ya sea por la idea de construir un modelo de ciudad digital, como por el gran número de centros de desarrollo con los que cuenta para la producción. Actualmente en varios de estos centros el estudio de los estándares establecidos por la ISO forma parte del ciclo de vida de las aplicaciones que desarrollan, uno de estos centros es el CISED.

El CISED es un centro dedicado a desarrollar productos y soluciones integrales en el campo de la identificación y la seguridad digital. Cuenta con cinco departamentos encargados de desarrollar diferentes líneas de investigación y desarrollo, entre los que se encuentra el Departamento de Seguridad Digital que cuenta con varios proyectos productivos, los cuales están actualmente en sus etapas iniciales de desarrollo. Uno de estos proyectos se desarrolla bajo la línea de investigación de estándares internacionales para el desarrollo de *software* y el estudio de los resultados de las pruebas de seguridad.

Actualmente en el Departamento de Seguridad Digital se cuenta con varias herramientas para realizar pruebas de seguridad a un *software*. El uso de estas herramientas hace posible detectar vulnerabilidades y fallas en las aplicaciones a las que se les realizan las pruebas. Conocer esta información no es suficiente, ya que los datos que se arrojan de las pruebas realizadas se pudieran procesar y de esta manera obtener información respecto al nivel de seguridad en las aplicaciones que puede ser aprovechada en el proyecto, contribuyendo así al proceso de toma de decisiones de los mismos.

INTRODUCCIÓN

Procesar la información generada de las pruebas de forma manual resulta engorroso debido a la gran cantidad de información que es arrojada. Por lo que para determinar el nivel de seguridad en las aplicaciones se requiere de una interpretación del resultado de las pruebas de manera automatizada, además del uso de métricas para medir los indicadores necesarios que hacen posible determinar la seguridad en la aplicación.

A partir de la situación problemática planteada, surge el siguiente **problema científico**: ¿Cómo mejorar el procesamiento de los datos arrojados por las pruebas de seguridad en las aplicaciones?

Por lo que se define como **objeto de estudio**: el proceso de evaluación de la seguridad en aplicaciones.

Para responder al problema de la investigación se tiene como **objetivo general:** desarrollar una herramienta que facilite el control y la evaluación de la seguridad en las aplicaciones que se desarrollan en el Departamento de Seguridad Digital, mediante el uso de métricas de seguridad.

Queda enmarcado como **campo de acción**: el proceso de evaluación de la seguridad en aplicaciones mediante el uso de métricas.

Para dar cumplimiento al objetivo general planteado anteriormente, se definieron un conjunto de **tareas de investigación** que se muestran a continuación:

1. Análisis de estándares internacionales de calidad y seguridad.
2. Análisis de métricas para la seguridad de un *software*.
3. Definición y aplicación de técnicas de recogida de información.
4. Análisis y explotación de herramientas para pruebas de seguridad.
5. Análisis del proceso de integración de datos ETL.
6. Diseño de la herramienta para la evaluación de la seguridad.
7. Implementación de la herramienta para la evaluación de la seguridad.
8. Realización de pruebas internas a la herramienta desarrollada.

Para el desarrollo de la investigación se utilizaron los siguientes métodos:

Teóricos:

- ✓ **Histórico-lógico:** se utiliza en la investigación para conocer si actualmente existen métodos de evaluación de seguridad usando los datos arrojados por herramientas de pruebas de seguridad y mediante el uso de métricas.
- ✓ **Analítico-sintético:** para analizar las diferentes maneras de evaluar la seguridad en las aplicaciones, realizando pruebas de seguridad y usando métricas definidas por estándares internacionales.

Empíricos:

- ✓ **Entrevista:** para conocer a través de los directivos del centro si se utiliza en el Departamento de Seguridad Digital algún método o herramienta para evaluar la seguridad en las aplicaciones.

Justificación de la investigación

El desarrollo de la investigación proporcionará en el Departamento de Seguridad Digital un modo de dar seguimiento a las aplicaciones que desarrollan, utilizando para ello la evaluación de la seguridad de las mismas. Debido a que existe gran necesidad de conocer el nivel de seguridad de estas aplicaciones, con el fin de tomar acciones correctivas a tiempo, que permitan al proyecto garantizar la entrega al cliente de un producto que cuente con la seguridad y calidad requerida, logrando superar todas sus expectativas.

El presente trabajo está estructurado en cuatro capítulos que a continuación se describen:

Capítulo 1: Fundamentación Teórica

El presente capítulo contiene las bases teóricas para entender el problema planteado. Se presentan los conceptos fundamentales relacionados con la seguridad de la información, la calidad y las métricas de seguridad. Se realiza un estudio de los diferentes estándares internacionales que existen para llevar a cabo el control y evaluación de la seguridad en aplicaciones. Además de ser estudiado el proceso de integración de datos ETL, utilizado para facilitar el trabajo en la implementación de la propuesta de solución.

Capítulo 2: Tecnologías, Herramientas y Metodología.

En este capítulo se analizan las tecnologías, herramientas y metodología que permitirán desarrollar la propuesta de solución a la problemática existente.

Capítulo 3. Características y Diseño del Sistema

En el capítulo se describe y caracteriza la solución propuesta partiendo de un análisis previo del objeto de estudio y la problemática planteada. Se realiza una especificación de las funcionalidades del sistema a través del levantamiento y descripción de los escenarios y los requisitos de calidad del servicio que debe cumplir la solución. Se especifica la arquitectura, así como los patrones de diseño a utilizar necesarios para implementar la solución.

Capítulo 4. Desarrollo y Estabilización del Sistema

En este capítulo se establecen los estándares de codificación a seguir para la implementación del sistema, se crean diagramas necesarios para guiar la implementación, y finalmente se realizan las pruebas necesarias para validar la solución propuesta.

CAPÍTULO 1

FUNDAMENTACIÓN TEÓRICA

1. Introducción

El presente capítulo contiene la investigación acerca de los conceptos esenciales para comprender los procesos asociados al problema planteado, el análisis de diferentes estándares para el desarrollo de aplicaciones seguras, así como el estudio de métricas de seguridad asociadas a estos. Se realiza además un estudio sobre el proceso de integración de datos ETL, utilizado para facilitar el trabajo en el desarrollo de la propuesta de solución.

1.1 Calidad de *software*

Actualmente controlar que el desarrollo de *software* se realice con la calidad requerida y contenga los elementos de seguridad necesarios para una buena ejecución, se ha convertido en uno de los elementos fundamentales en la producción de *software*.

En 1992 Roger S. Pressman define la calidad como: “Concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos con los estándares de desarrollo explícitamente documentados y con las características implícitas que se espera de todo *software* desarrollado profesionalmente”. (3)

En 1998 Pressman plantea otra definición: “Concordancia del *software* producido con los requerimientos explícitamente establecidos, con los estándares de desarrollo prefijados y con los requerimientos implícitos no establecidos formalmente, que desea el usuario”. (4)

Por su parte la ISO establece diferentes definiciones para la calidad del *software* en cada uno de sus estándares, ejemplo de ellos:

Estándar ISO/IEC 8402:1994, define el término calidad como: “Totalidad de propiedades y características de un producto, proceso o servicio que le confiere su aptitud para satisfacer unas

necesidades expresadas o implícitas.” (5) También en el estándar ISO/IEC 9000:2000, se define calidad como: “Grado en el que un conjunto de características inherentes cumple con los requisitos”. En esta definición se hace especial énfasis en cumplir los requerimientos de los usuarios. (5)

Después de haber analizado varios de los criterios y definiciones, los autores llegan a la conclusión de que la calidad es un área importante a la que se recomienda dedicar esfuerzo y tiempo, ya que la misma se puede considerar como un conjunto de cualidades que caracterizan al *software* y determinan su utilidad. Calidad es sinónimo de eficiencia, corrección, seguridad y confiabilidad, elementos que combinados pueden contribuir a lograr un producto que se corresponda con las necesidades y expectativas del cliente.

1.2 Seguridad informática

Con el fin de garantizar que todo aquello que represente un elemento de vital importancia esté debidamente protegido, surge una palabra clave para el desarrollo de la humanidad: seguridad. La seguridad de la información específicamente, engloba todas aquellas medidas preventivas y reactivas del hombre, de las organizaciones y de los sistemas tecnológicos que permitan resguardar y proteger la información buscando mantener la confidencialidad, la disponibilidad e integridad de la misma. (6)

En el mundo digital existe estrecha relación entre la seguridad de la información y la seguridad informática, esto se debe a que esta última como disciplina se encarga de proteger la integridad y la privacidad de la información almacenada en un sistema informático. Dichos sistemas pueden ser protegidos desde un punto de vista de seguridad física o lógica. La primera se basa fundamentalmente en los daños que pudieran recibir los componentes del *hardware*; y la segunda se refiere al tema de la protección del *software* y de la información que estos contienen, a través de una arquitectura de seguridad eficiente.

Un sistema seguro debe ser *íntegro,* es decir, la información solo puede ser modificada por las personas autorizadas; *confidencial,* porque los datos tienen que ser legibles únicamente para los usuarios autorizados; *irrefutable,* porque el usuario no debe poder negar las acciones que realizó y tener buena *disponibilidad*, debe ser *estable*. En otras palabras, puede decirse que la seguridad informática busca

garantizar que los recursos de un sistema de información sean utilizados tal como una organización o un usuario lo ha decidido, sin intromisiones.

1.3 Estándares internacionales

Para garantizar el éxito en la gestión de la información de todo producto, por considerar que forma parte de los activos fundamentales de todo sistema, se toman en cuenta los elementos que brindan los estándares para la seguridad. Un estándar se define como el grado de cumplimiento exigible a un criterio de calidad. Dicho en otros términos, define el rango en el que resulta aceptable el nivel de calidad que se alcanza en un determinado proceso. (7)

Los estándares de calidad determinan el nivel mínimo y máximo aceptable para un indicador. Si el valor del indicador se encuentra dentro del rango significa que se cumple con el criterio de calidad definido y que las cosas transcurren conforme a lo previsto. (7)

Los estándares de seguridad por su parte, son considerados como guías de buenas prácticas, que contribuyen a lograr un correcto diseño de la planificación que se va a realizar para el desarrollo de un producto. Esto con el objetivo de resguardar y proteger la información buscando mantener la confidencialidad, la disponibilidad e integridad de la misma. Además de lograr que el *software* contenga una infraestructura en lo posible fuerte y segura.

1.3.1 Estándar ISO/IEC 9126

El estándar ISO/IEC 9126 tuvo su primera versión en el año 1991, posteriormente fue revisado y actualizado en el 2001. Fue creado para la evaluación del *software* estableciendo un modelo de calidad, es supervisado por el proyecto Requisitos y Evaluación de Calidad de Productos de *Software* (SQuaRE) y pensado para los desarrolladores. Su estructura se encuentra dividida en cuatro partes las cuales se distribuyen de la siguiente manera:

- ✓ ISO/IEC 9126-1 Modelo de calidad.
- ✓ ISO/IEC 9126-2 Métricas externas.
- ✓ ISO/IEC 9126-3 Métricas internas.

- ✓ ISO/IEC 9126-4 Métricas de calidad en el uso.

La ISO/IEC 9126-1 propone una serie de características y sub-características de calidad para ser tomadas en cuenta y evaluadas en un producto. Entre las características se definen:

- ✓ Funcionalidad, Fiabilidad, Usabilidad, Eficiencia, Mantenibilidad y Portabilidad. (2)

Para cada una de estas características se definen a su vez, una serie de sub-características que responden a cada una de ellas. Esencialmente, según el tema principal a tratar en el presente trabajo se tendrá en cuenta la sub-característica seguridad, que pertenece a la característica funcionalidad dentro de la cual se evalúa el grado en que el *software* satisface las necesidades.

1.3.2 Estándar ISO/IEC 14598

Este estándar indica los requisitos a tener en cuenta para la aplicación de los métodos de medición y para el proceso de evaluación, proporcionando un entorno de trabajo para la evaluación de la calidad de diferentes tipos de productos *software*. La ISO/IEC 14598 consta de seis partes que especifican el proceso a seguir para evaluar *software*:

- ✓ ISO/IEC 14598-1 Visión general.
- ✓ ISO/IEC 14598-2 Planificación y Gestión.
- ✓ ISO/IEC 14598-3 Procedimiento para desarrolladores.
- ✓ ISO/IEC 14598-4 Procedimiento para compradores.
- ✓ ISO/IEC 14598-5 Procedimiento para evaluadores.
- ✓ ISO/IEC 14598-6 Documentación de los módulos de evaluación. (8)

La quinta parte del estándar (ISO/IEC 14598-5) se ajusta a las necesidades actuales de los autores, ya que especifica y explica el procedimiento por el cual los evaluadores del *software* se pueden guiar para desarrollar este proceso de evaluación. Esta parte del estándar proporciona requisitos y recomendaciones para la implantación práctica de la evaluación del producto *software*. Puede ser usada para aplicar los conceptos descritos en el ISO/IEC 9126.

Las actividades de evaluación son una característica fundamental para el desarrollo efectivo del *software*, se necesita tener cronogramas de evaluación que brinden información óptima durante el ciclo de vida del producto. Los elementos necesarios a tener en cuenta para una correcta evaluación son componentes claves de la ISO/IEC 14598. (9)

1.3.3 Estándar ISO/IEC 25000

Estándar que surge en el año 2005, siendo actualizado en mayo de 2010. Fue creado con el fin de integrar en una sola familia el estándar ISO/IEC 9126 y el ISO/IEC 14598, es decir, agrupar el modelo de calidad y el proceso de evaluación.

La familia ISO 25000 tiene como principales objetivos los siguientes:

1. Guiar el desarrollo de los productos de *software* además de establecer criterios para la especificación de requisitos de calidad y métricas.
2. Establecer un modelo de calidad para el producto de *software*, definiendo las características y sub-características que se deben tener en cuenta.
3. Definir el proceso de evaluación de la calidad del producto de *software*, con el fin de poder establecer la calidad en función del modelo.

Este estándar proporciona además una guía para el uso de las nuevas series de estándares internacionales llamados SQuaRE con el fin de lograr organizar, enriquecer y unificar las series que cubren dos procesos principales: especificación de requerimientos de calidad del *software* y evaluación de la calidad del *software*, soportada por el proceso de medición de calidad del *software*. (9)

1.3.4 Estándar ISO/IEC 27001

El estándar ISO/IEC 27001 surge en el año 2005 y es conocido como un estándar eficaz y potente para gestionar la seguridad de forma óptima. Esto se debe a que establece métricas e indicadores de seguridad para evaluar la calidad de los productos de *software*. Contemplándolos como un accesorio de gran importancia que se debe añadir al proceso de desarrollo de *software* a lo largo de su ciclo de vida, garantizando así, que el producto sea revisado y mejorado de forma permanente.

La utilización de este estándar permite definir cuáles son las métricas que realmente interesa poner en marcha según el tipo de *software* con el que se está trabajando, cómo implementarlas y de qué manera mejorarlas a través del establecimiento de un ciclo denominado Planificar-Hacer-Revisar-Actuar (PDCA) en el cual se describen paso a paso el proceso de selección. (10)

Después de analizar cada uno de los estándares y debido a que poseen elementos que se corresponden con las características que debe poseer el sistema a desarrollar, se decidió utilizar lo establecido por el ISO/IEC 25000 en cuanto a elementos de calidad y métricas, así como los pasos necesarios para una correcta selección de métricas de seguridad establecidos por el ISO/IEC 27001.

1.4 Métricas de seguridad

Las métricas de seguridad son definidas como "La aplicación continua de mediciones basadas en técnicas para el proceso de desarrollo de *software* y sus productos, para suministrar información relevante a tiempo, así el administrador junto con el empleo de estas técnicas mejorará el proceso y sus productos". (11)

Las métricas deben ser un instrumento que ayude a mejorar el desarrollo de aplicaciones informáticas, no tiene mucho sentido aplicar métricas que lejos de ayudar a los desarrolladores constituyan un problema; bien por ser demasiado complejas, porque no se entiendan correctamente los objetivos que persiguen o porque arrojen resultados imprecisos que no puedan ser interpretados.

En la actualidad el uso de las métricas en el amplio mercado del *software* se está poniendo en práctica cada vez más y con gran éxito. Esto se debe a que los desarrolladores han reconocido su importancia para valorar y gestionar de forma efectiva la calidad con que son realizados los productos. Las métricas contribuyen en gran medida a tener con mayor exactitud una noción del esfuerzo, tiempo de desarrollo y posibles errores que presenta el *software* analizado, ayudando a tener conocimiento acerca del nivel de madurez y seguridad con que contará el producto final.

Las métricas son clasificadas según el tipo de pruebas para las que estén creadas. Para la propuesta de solución planteada serán analizadas las métricas de seguridad, las cuales se definen como el

conjunto de preceptos y reglas, necesarios para poder medir de forma real el nivel de seguridad de un sistema. (12)

Para lograr una correcta selección de las métricas a utilizar se deberá tener en cuenta algunos aspectos que se detallan a continuación:

¿Para qué? (se quiere medir):

- ✓ Medir la evolución de la seguridad de un sistema o producto.
- ✓ Tener datos elocuentes sobre el estado de seguridad de un producto.
- ✓ Saber dónde hay que hacer hincapié en la mejora de la seguridad.

¿Qué? (se quiere medir):

- ✓ Cantidad y tipo de amenazas.
- ✓ Vulnerabilidades y puntos débiles.

¿Cómo? (se puede medir):

- ✓ ¿Cómo convertir la información en datos válidos?
- ✓ ¿Qué fuentes y herramientas se necesitan? (12)

La dificultad en la definición de una métrica está precisamente en concretar estos tres factores. Una vez que se especifiquen estos aspectos, entonces se podrá pasar al proceso de selección o creación de las métricas a ser aplicadas para la evaluación. Como se puede observar a continuación, cada métrica debe tener un propósito, un método de aplicación, una fórmula y finalmente, una manera de interpretar el valor obtenido después de calculada.

Métrica	Propósito	Método de Aplicación	Fórmula	Interpretación del valor obtenido
Detección de vulnerabilidades	Determinar cantidad de vulnerabilidades	Contar el número de vulnerabilidades del sistema y comparar	**X = B/A** A: Número de vulnerabilidades	**0 <= X <= 1** A mayor cercanía al 1

	de seguridad del sistema.	con el valor mínimo requerido para tener una adecuada seguridad en el sistema.	detectadas en el sistema. B: Valor mínimo requerido de vulnerabilidades. (13)	resultará más seguro.
Identificación de riesgos	Identificar posibles riesgos de seguridad que pudieran afectar el funcionamiento del sistema en determinado momento.	Contar la cantidad de riesgos detectados.	**X = 1- A/B** A: Número de riesgos que pudieran convertirse en vulnerabilidades. B: Número de riesgos detectados en el sistema.	**0 <= X <= 1** A mayor cercanía al 1 menor probabilidad de que los riesgos se conviertan en vulnerabilidad.
Controlabilidad de acceso	¿Cuán controlable es el acceso al sistema?	Determinar posibles puntos que permitan realizar inyecciones SQL con el objetivo de violar el acceso al sistema y obtener información valiosa.	**X =** Cantidad de puntos que permiten realizar inyecciones SQL.	Mientras menor más seguro.

Tabla 1: Métricas de seguridad aplicables al sistema

Después de aplicadas las métricas, entonces es posible obtener una serie de valores que son llevados a una escala (**Ver Tabla 2**), y que definen un nivel muy alto, alto, medio o bajo de seguridad en el *software*.

Intervalo	Nivel
0 - 0,4	Bajo
0,4 - 0,7	Medio
0,7 - 0,9	Alto
0,9 - 1	Muy Alto

Tabla 2: Escala para la evaluación

1.5 Herramienta de pruebas de seguridad

Actualmente existen muchas herramientas para realizar pruebas de seguridad con el fin de encontrar vulnerabilidades en el *software*, y de esta manera garantizar la seguridad y la calidad del mismo. La herramienta que más se ajusta a las expectativas que se tienen es Web Application Attack and Audit *Framework* (W3AF) por ser la más integradora de las herramientas de pruebas, además permite realizar diferentes tipos de pruebas de seguridad.

La herramienta W3AF es un *framework* libre, que permite realizar diferentes tipos de pruebas de seguridad a aplicaciones web para determinar sus vulnerabilidades. Es de fácil uso y aporta resultados a los cuales se les puede aplicar métricas para así determinar el nivel de seguridad de una aplicación. Otro de los elementos que le atribuyen importancia a esta herramienta, es su capacidad de no limitarse simplemente a emitir los resultados de un escaneo para que sean almacenados como vulnerabilidades, sino que realiza el escaneo de un objetivo, encuentra las vulnerabilidades en el mismo y es capaz de explotarlas. (14)

1.6 Proceso de integración de datos ETL

ETL son las siglas en inglés de Extraer, Transformar y Cargar (*Extract, Transform and Load*). Este es un proceso de gran importancia cuando de agilizar el trabajo se trata, además de ahorrar tiempo, sencillamente porque permite mover datos desde múltiples fuentes, reformatearlos, limpiarlos, y cargarlos en una BD o en otro sistema operacional para analizarlos y obtener de ellos solo lo necesario para trabajar.

Cada uno de las acciones que representan las siglas ETL se ejecutan de forma continua para lograr tener éxito en el proceso. A continuación se realiza una breve descripción de cada una de estas:

Extracción

Esta es la primera parte del proceso ETL, consiste en extraer los datos desde los sistemas de origen. La mayoría de las fuentes de almacenamiento de datos fusionan datos provenientes de diferentes sistemas de origen.

Cada sistema separado puede usar una fuente diferente de datos o formatos distintos. Los formatos de las fuentes normalmente se encuentran en bases de datos relacionales o ficheros planos, pero pueden incluir bases de datos no relacionales u otras estructuras diferentes. La extracción convierte los datos a un formato preparado para iniciar el proceso de transformación.

Transformación

La transformación es el elemento básico de diseño de los procesos ETL. Una transformación se compone de pasos, que están enlazados entre sí a través de los saltos. Los pasos son el elemento más pequeño dentro de las transformaciones y los saltos constituyen el elemento a través del cual fluye la información entre los diferentes pasos (siempre es la salida de un paso y la entrada de otro).

La fase de transformación aplica una serie de reglas de negocio o funciones sobre los datos extraídos para convertirlos en datos que serán cargados. Algunas fuentes de datos requerirán alguna pequeña manipulación de los datos. No obstante, en otros casos pueden ser necesarias aplicar algunas de las siguientes reglas:

- ✓ Seleccionar solamente ciertas columnas para su carga (por ejemplo, que las columnas con valores nulos no sean cargadas).
- ✓ Traducir códigos (por ejemplo, si la fuente almacena una “H” para Hombre y “M” para Mujer pero el destino tiene que guardar “1” para Hombre y “2” para Mujer).
- ✓ Codificar valores libres (por ejemplo, convertir “Hombre” en “H” o “Sr” en “1”).
- ✓ Unir datos de múltiples fuentes (por ejemplo, búsquedas y combinaciones, entre otros).

- ✓ Dividir una columna en varias (por ejemplo, columna “Nombre: García, Miguel”; pasar a dos columnas “Nombre: Miguel” y “Apellido: García”).

Carga

La fase de carga es el momento en el cual los datos de la fase anterior son cargados en el sistema de destino. Dependiendo de los requerimientos de la organización, este proceso puede abarcar una amplia variedad de acciones diferentes. En algunas BD se sobrescribe la información antigua con nuevos datos.

Esta fase interactúa directamente con la BD de destino. Al realizar esta operación se aplicarán todas las restricciones y disparadores (triggers) que se hayan definido (por ejemplo, valores únicos, campos obligatorios, rangos de valores). Estas restricciones y disparadores (si están bien definidos) contribuyen a que se garantice la calidad de los datos en el proceso ETL, y deben ser tenidos en cuenta. (15)

Este proceso formará parte del desarrollo del sistema propuesto con el objetivo de hacer menos complejo el trabajo de los desarrolladores, debido a que es preciso seleccionar de la gran cantidad de información que proporcionarán los resultados arrojados por las pruebas de seguridad, solo los elementos necesarios para desarrollar la evaluación.

1.7 Conclusiones del capítulo

El estudio del estado del arte permitió contar con los conocimientos suficientes para dar solución a la problemática existente, teniendo como base teórica para la investigación los métodos teóricos y empíricos definidos inicialmente. Entre los referentes estudiados se encuentran los estándares internacionales, los cuales permitieron realizar una selección de las métricas de seguridad que se consideran más adecuadas para el proceso de evaluación de la seguridad de un *software*. Finalmente, dada la necesidad de optimizar tiempo y esfuerzo para el desarrollo de la solución que se desea proporcionar, se estudió detalladamente el proceso de integración de datos ETL.

TECNOLOGÍAS, HERRAMIENTAS Y METODOLOGÍA

2. Introducción

En el presente capítulo se describen todas las tecnologías, herramientas y diferentes tipos de lenguajes a utilizar para el desarrollo de la propuesta de solución. Además, se caracteriza la metodología que guiará el proceso de desarrollo del sistema, permitiendo elaborar los artefactos correspondientes a las mismas, así como planificar cada una de las actividades a desarrollar. La selección de cada uno de estos elementos para el desarrollo de la propuesta viene dado en parte, por la decisión del proyecto que actualmente se desarrolla en el Departamento de Seguridad Digital.

2.1 Análisis de metodologías de desarrollo de *software*

El desarrollo de *software* en ocasiones es un poco riesgoso y difícil de controlar, pero si no llevamos una metodología de por medio, es posible que se obtenga un producto final que no era el esperado inicialmente, lo que traería consigo clientes insatisfechos con los resultados.

Existen dos tipos de metodologías, las robustas y las ágiles, siendo esta última el tipo de metodología establecido que se use en el Departamento de Seguridad Digital del CISED. Las metodologías ágiles surgen como una extensión a las metodologías tradicionales para mejorar el desarrollo de sistemas, según el tipo de proyecto y empresa, en otras palabras surgen para optimizar las prácticas de desarrollo de *software*. Por tal motivo se hace necesario definir metodologías para guiar el proceso de desarrollo de un producto de *software*.

Las metodologías se definen según los pasos a seguir para el cumplimiento de un objetivo. El objetivo fundamental dentro del desarrollo de *software* es producir un producto con la calidad requerida del cliente que cumpla con los requerimientos.

Entre las metodologías ágiles más usadas en el desarrollo de un *software* se destacan:

- ✓ XP (eXtreme Programming)
- ✓ MSF (Microsoft Solution *Framework*)
- ✓ Scrum

2.1.1 Programación Extrema (Extreme Programming, XP)

Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de *software*, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo.

XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. Es utilizada para proyectos de corto plazo y equipos pequeños. La metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto.

XP tiene como características principales que se basa en:

- ✓ **Pruebas Unitarias:** Se basa en las pruebas realizadas a los principales procesos, de tal manera que adelantándonos en algo hacia el futuro, podamos hacer pruebas de las fallas que pudieran ocurrir. Es como adelantarse a obtener los posibles errores.
- ✓ **Refabricación:** Se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio.
- ✓ **Programación en pares:** Una particularidad de esta metodología es que propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento. Es como el chofer y el copiloto mientras uno conduce, el otro consulta el mapa. (16)

2.1.2 Microsoft Solution *Framework* for Agile *Software* Development

MSF Agile es la nueva propuesta de Microsoft en el mundo de procesos y prácticas ágiles de desarrollo de *software*. "Microsoft Agile" renueva al ya exitoso MSF V3.0, que es un marco de trabajo en cascada y espiral, implementando las mejores prácticas del mundo de desarrollo ágil de *software*.

Esta metodología tiene como principales características el ser de planificación adaptable a los cambios y enfocada a las personas. La misma incorpora prácticas para el manejo de la calidad del servicio (el rendimiento y la seguridad) y facilita la automatización y la orientación que se necesita para apoyar el equipo de trabajo, incluyendo la gestión de configuración y de proyectos. Su proceso se desarrolla mediante la construcción de aplicaciones basadas en escenarios, que no son más que las funcionalidades que debe realizar el *software*.

Cómo metodología cuenta con una serie de principios para su aplicación que a continuación se enuncian:

- ✓ Formar equipo con el cliente.
- ✓ Promocionar las comunicaciones abiertas.
- ✓ Trabajar con una visión común.
- ✓ La calidad es el negocio de todos, todos los días.
- ✓ Mantenerse ágil, adaptarse a los cambios.
- ✓ Hacer del despliegue un hábito.
- ✓ Crear un flujo de valor.

Esta metodología incluye cinco fases para el desarrollo y seguimiento del producto, estas son: Visión y Alcance, Planificación, Desarrollo, Estabilización y Despliegue.

MSF Agile dispone de los elementos de trabajo siguientes:

- ✓ **Escenario:** descripción de la necesidad o solicitud del usuario.
- ✓ **Error:** defecto o desviación entre el comportamiento esperado y el comportamiento observado en el producto.

- ✓ **Requisitos de calidad del servicio:** material resultante esperado del producto final. El mismo puede ser un resultado, un problema resuelto o una característica, entre otros.
- ✓ **Tarea:** acción independiente que debe realizar una persona o un grupo de personas.
- ✓ **Riesgo:** evento o condición probable que puede dar resultados potencialmente negativos en el proyecto en el futuro. (17)

Esta metodología tiene un diseño óptimo para proyectos pequeños con un calendario de entrega rápido por lo que puede ser conveniente elegir cuando:

- ✓ No se necesita muchos procesos documentados.
- ✓ Los equipos de desarrollo de *software* son reducidos.
- ✓ Existen ciclos de desarrollo de *software* cortos (medidos en semanas o meses).

La metodología de desarrollo que se definió para guiar el proceso de desarrollo del *software* a construir, es MSF for ASD. El uso de esta metodología está establecido como política del proyecto para el desarrollo del sistema, la misma permitirá guiar todo el ciclo de vida, proporcionando elementos en cada una de las fases que serán de gran ayuda para las fases posteriores. De forma general para este sistema, MSF Agile representará el eje alrededor del cual girarán todos los componentes necesarios para conformar el producto final, añadiendo elementos que se consideren necesarios para una mejor comprensión del mismo.

2.2 Lenguaje de modelado

El lenguaje de modelado seleccionado para guiar el diseño de la aplicación fue el Lenguaje Unificado de Modelado (por sus siglas en inglés UML). UML es un grupo de especificaciones de notación orientadas a objeto, las cuales están compuestas por distintos diagramas, que representan las diferentes etapas del desarrollo de un proyecto de *software*. Ofrece además un estándar para describir un modelo del sistema, incluyendo aspectos conceptuales tales como procesos de negocio, funciones del sistema, aspectos concretos como expresiones de lenguajes de programación, esquemas de BD y componentes reutilizables. Este lenguaje indica cómo crear y leer los modelos, pero no dice cómo desarrollarlos. Los objetivos de UML son muchos, pero se pueden sintetizar sus funciones:

- ✓ **Visualizar:** permite expresar gráficamente un sistema de forma que otra persona lo pueda entender.
- ✓ **Especificar:** permite especificar las características de un sistema antes de su construcción.
- ✓ **Construir:** a partir de los modelos especificados se pueden construir los sistemas diseñados.
- ✓ **Documentar:** los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para su futura revisión. (18)

El lenguaje UML será utilizado para guiar el diseño de los diagramas que contribuyan a comprender como va a funcionar el sistema y de qué manera debe ser implementado para lograr un correcto funcionamiento.

2.3 Herramientas CASE

Las herramientas CASE son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de *software* reduciendo el coste de las mismas en términos de tiempo y de dinero. Estas herramientas pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del *software* en tareas como el proceso de realizar el diseño del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación y detección de errores. (19)

2.3.1 Visual Paradigm

Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de *software*: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. También proporciona abundantes tutoriales de UML, demostraciones interactivas y proyectos. Presenta licencia gratuita y comercial. Es fácil de instalar, actualizar y compatible entre ediciones. (20)

Entre las principales características de Visual Paradigm se encuentran:

- ✓ Entorno de creación de diagramas para UML.

- ✓ Diseño centrado en casos de uso y enfocado al negocio que generan un *software* de mayor calidad.
- ✓ Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- ✓ Disponibilidad en múltiples plataformas.

2.3.2 Altova UModel

La herramienta de modelado Altova UModel tiene como propósito crear e interpretar diseños de *software* mediante el uso del lenguaje UML descrito en el epígrafe anterior, la misma representa el punto de partida para el desarrollo de *software* exitoso. En su versión Enterprise Edition v2010, presenta un entorno de trabajo agradable, sencillo a la vista y muy intuitivo a la hora de realizar los diferentes modelos.

UModel es una herramienta rápida, eficaz y práctica para todos los programadores y gestores de proyecto. Su rica interfaz visual y usabilidad superior ayudan a incrementar el aprendizaje en términos de UML, permitiendo que desarrolladores, incluso los nuevos, manejen rápidamente este lenguaje de modelado. Su uso potencia el incremento en la productividad y maximiza la calidad en los resultados debido a que solventa los problemas velozmente. De manera sencilla UModel permite mantener el proyecto sincronizado y al día. (21)

Esta herramienta será utilizada para diagramar la relación entre los elementos del sistema, así como para modelar la BD correspondiente y representar el despliegue de la herramienta a desarrollar.

2.4 Servidor web

Un servidor web brinda contenido estático a un navegador, carga un archivo y lo envía a través de la red al navegador de un usuario. Este intercambio es mediado por el navegador y el servidor que hablan el uno con el otro mediante el protocolo HTTP (Hypertext Transfer Protocol).

Se define como servidor web a utilizar Apache 2.0, debido a que es flexible, rápido y eficiente, continuamente actualizado y adaptado a los nuevos protocolos. Está desarrollado por la Fundación de

Software Apache (Apache *Software* Foundation), es una tecnología gratuita de código fuente abierto con licencia descendiente de las licencias BSD[2], por lo que se puede hacer uso de su código fuente.

Lo que hace a este servidor web universal, es que se puede ejecutar en varios sistemas operativos. Es una tecnología altamente configurable de diseño modular. Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. Es posible configurar Apache para que ejecute un determinado *script* cuando ocurra un error en concreto. Brinda soporte para varios lenguajes: PHP, JAVA, Perl y librerías ASP.

Entre sus principales características se destaca que:

- ✓ Es un servidor web conforme al protocolo HTTP.
- ✓ El diseño modular de Apache permite a los administradores de sitios web elegir qué características se van a incluir en el servidor al seleccionar los módulos que se van a cargar, ya sea al compilar o al ejecutar el servidor.
- ✓ Incentiva la realimentación de los usuarios, obteniendo nuevas ideas, informes de fallos y parches para la solución de los mismos. (22)

Apache como servidor web albergará el Sistema de evaluación de seguridad y se encontrará a la espera de que un usuario realice alguna petición, mediante esta solicitud publicará el contenido deseado mediante el protocolo HTTPS.

2.5 Sistema de gestión de contenidos

Un sistema de gestión de contenidos (Content Management System o CMS) es un sistema que permite crear una estructura de soporte (*framework*) para la creación y administración de contenidos, en páginas web ya sea en Internet o en una Intranet, por lo que son conocidos también como gestores de contenido web (Web Content Management o WCM).

Entre sus principales características se tiene que:

[2] Por sus siglas en inglés Berkeley *Software* Distribution.

- ✓ Permite que sin conocimientos de programación cualquier usuario pueda añadir contenido en el portal web.
- ✓ Puede adaptarse a las preferencias o necesidades de cada usuario.
- ✓ Puede proporcionar compatibilidad con los diferentes navegadores disponibles en todas las plataformas y su capacidad le permite adaptarse al idioma y cultura del visitante.

Los CMS más utilizados en la actualidad son: Joomla y Drupal.

2.5.1 Joomla

Joomla es un CMS de código abierto programado en lenguaje PHP bajo licencia GNU/GPL y que utiliza como gestor de BD para el almacenamiento del contenido y los parámetros de configuración del sitio MySQL. Es un sistema gestor de contenidos dinámico que permite crear sitios web de alta interactividad, profesionalidad y eficiencia tanto para noticias como para sitios corporativos y portales comunitarios. La administración de Joomla está enteramente basada en la gestión en línea de contenidos. Posee una interfaz amigable y sencilla para cualquier tipo de usuario.

Como principales características de este CMS tenemos las siguientes:

- ✓ Permite convertir una web estática en un portal con diferentes funcionalidades y características dinámicas.
- ✓ Facilita la introducción y actualización de contenidos.
- ✓ Es un sistema administrado.
- ✓ Permite la restricción al acceso a determinados contenidos a usuarios con permisos especiales.
- ✓ Diseño y contenido se manejan de forma independiente. (23)

2.5.2 Drupal

Drupal es un sistema de gestión de contenido modular y configurable, escrito en PHP, desarrollado y mantenido por una activa comunidad de usuarios. Destacada por la calidad de su código y de las páginas generadas, el respeto de los estándares de la web, y un énfasis especial en la usabilidad y

consistencia de todo el sistema. Su flexibilidad y adaptabilidad, así como la gran cantidad de módulos adicionales disponibles hace que sea adecuado para realizar diferentes tipos de sitio web.

Drupal es un CMS que tiene varias características que lo hacen sencillo y fácil de entender, dichas características se exponen a continuación:

- ✓ **Permisos basados en roles:** los administradores de Drupal no tienen que establecer permisos para cada usuario. En lugar de eso, pueden asignar permisos a un rol y agrupar los usuarios por roles.
- ✓ **Plantillas:** el sistema de temas de Drupal separa el contenido de la presentación permitiendo controlar o cambiar fácilmente el aspecto del sitio web. Se pueden crear plantillas con HTML y PHP.
- ✓ **Ayuda *on-line*:** posee un sistema de ayuda online muy eficaz y rápido.
- ✓ **Código abierto:** al contrario de otros sistemas de *blogs* o de gestión de contenido propietarios, es posible extender o adaptar Drupal según las necesidades reales de cada usuario.
- ✓ **Módulos:** la comunidad de Drupal ha contribuido con muchos módulos que proporcionan funcionalidades como “página de categorías”, autenticación mediante LDAP (Lightweight Directory Access Protocol, en español Protocolo Ligero de Acceso a Directorios), mensajes privados, foros, entre otros. (24)

Fue seleccionado para dar solución al problema planteado el CMS Drupal, debido a que es de código abierto, distribuido con licencia GNU/GPL y se encuentra en constante actualización, lo que hacen de este un CMS eficiente y abarcador.

2.6 Lenguajes de programación para el desarrollo web

Un lenguaje de programación es un conjunto de instrucciones, órdenes, comandos y reglas que permite la creación de programas. Estos permiten al programador indicar lo que debe hacer un programa mediante instrucciones que resultan ser comprensibles por las personas. Los lenguajes representan en forma simbólica y en manera de un texto, los códigos que ejecuta la computadora. (25)

Para el desarrollo de aplicaciones web, se emplean diferentes y numerosos lenguajes, los cuales se diferencian entre sí en dependencia de las características específicas del producto final que se quiera obtener.

2.6.1 Lenguaje del lado del servidor

Para el desarrollo de la solución se decidió utilizar como lenguaje de programación del lado del servidor PHP en su versión 5.3, ya que es un lenguaje implícito en el funcionamiento del CMS Drupal. A continuación se exponen algunas de las características del mismo para una mejor comprensión de por qué su selección.

PHP constituye un lenguaje *script* de alto nivel interpretado del lado del servidor. Es un lenguaje de programación (originario del nombre PHP Tools o Personal Home Page Tools) que sirve principalmente para proporcionar características dinámicas a una página web. Al ser ejecutado del lado del servidor, permite acceder a los recursos internos del mismo o a otros externos, como por ejemplo a una BD, siendo el resultado normalmente una página HTML con los datos y acciones enviadas al cliente.

A continuación se exponen algunas de sus ventajas:

- ✓ Es un lenguaje multiplataforma.
- ✓ Completamente orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una BD.
- ✓ Capacidad de conexión con la mayoría de los motores de BD que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.
- ✓ El código fuente escrito en PHP es invisible al navegador y al cliente, es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable.
- ✓ Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- ✓ Ofrece las herramientas necesarias para desarrollar aplicaciones complejas, utilizando orientación a objetos, patrones de diseño y otras técnicas avanzadas. (26)

2.6.2 Lenguajes del lado del cliente

CSS

Es el acrónimo de Cascading Style Sheets (es decir, hojas de estilo en cascada). Es un lenguaje de estilo usado para definir la presentación de un documento estructurado escrito en HTML o XML. La idea que se encuentra detrás del desarrollo de CSS es separar la estructura de un documento de su presentación. Las hojas de estilo abarcan cuestiones relativas a fuentes, colores, márgenes, líneas, altura, ancho, imágenes de fondo, posicionamiento avanzado y muchos otros temas.

Las ventajas de utilizar CSS son:

- ✓ Control centralizado de la presentación de un sitio web completo con lo que se agiliza de forma considerable la actualización del mismo.
- ✓ Los navegadores permiten a los usuarios especificar su propia hoja de estilo local que será aplicada a un sitio web, con lo que aumenta considerablemente la accesibilidad. Por ejemplo, personas con deficiencias visuales pueden configurar su propia hoja de estilo para aumentar el tamaño del texto o remarcar más los enlaces.
- ✓ Una página puede disponer de diferentes hojas de estilo según el dispositivo que la muestre o incluso a elección del usuario.
- ✓ El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño (siempre y cuando no se utilice estilo en línea). (27)

HTML

El Lenguaje de Marcas de Hipertexto (Hyper Text Markup Language) es el lenguaje con el que se desarrollan las páginas web. Permite escribir texto de forma estructurada, es decir, es un lenguaje de hipertexto y está compuesto por etiquetas, que marcan el inicio y el fin de cada elemento del documento. Un documento hipertexto no se compone únicamente de texto, sino que también contiene imágenes, sonido, vídeos, por lo que el resultado puede considerarse como un documento multimedia. Los documentos HTML deben tener la extensión html o htm, para que puedan ser visualizados en los

navegadores. Los navegadores se encargan de interpretar el código HTML de los documentos, y de mostrar a los usuarios las páginas web resultantes del código interpretado. (28)

2.7 Entorno Integrado de Desarrollo

Un Entorno Integrado de Desarrollo (IDE por sus siglas en inglés Integrated Development Environment), es un conjunto de herramientas que han sido creadas como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica.

Un IDE provee un marco de trabajo poco complejo para la mayoría de los lenguajes de programación y en algunos casos puede funcionar como un sistema en tiempo de ejecución en donde se permite utilizar el lenguaje de programación en forma interactiva, sin necesidad de trabajo orientado a archivos de texto. Entre los entornos integrados de desarrollo más utilizados se encuentran Eclipse, Zend Studio y NetBeans.

2.7.1 Zend Studio

Zend Studio o Zend Development Environment es un completo entorno integrado de desarrollo para el lenguaje de programación PHP. Está escrito en Java, y es multiplataforma, disponible para Microsoft Windows, Mac OS X y GNU/Linux. Soporta PHP4 y PHP5, presenta plegado de código, inserción automática de paréntesis y corchetes de cierre, detección de errores de sintaxis en tiempo real entre otras muchas funcionalidades que agilizan el trabajo del programador. Incluye además funciones de errores de depuración permitiendo dicha acción en servidores remotos. Soporte para la gestión de grandes proyectos, para el control de versiones y para la navegación en BD y ejecución de consultas SQL. (29)

2.7.2 NetBeans

NetBeans IDE es un entorno de desarrollo integrado libre sin restricciones de uso. Fue creado principalmente para el lenguaje de programación Java, pero puede servir para otros lenguajes de programación entre ellos PHP 5. Mediante el que permite crear aplicaciones web o componentes de alta calidad que pueden ser integrados a CMS como Drupal.

A continuación se detallan algunas de las principales características de este IDE:

- ✓ El proyecto NetBeans ofrece una versión del IDE a la medida para el desarrollo en PHP de sitios web que abarcan una variedad de secuencias de comandos y lenguajes de marcado.
- ✓ El IDE NetBeans respalda plenamente el desarrollo iterativo, por lo que los proyectos desarrollados en PHP siguen los patrones clásicos conocidos para los desarrolladores web.
- ✓ El editor de NetBeans para PHP ofrece plantillas de código, información sobre herramientas de parámetros, consejos y soluciones rápidas (aplicación de métodos abstractos), autocompletado de código inteligente (incluido el soporte de terminación), formateo de código y marcado de los sucesos y los puntos de salida.
- ✓ El editor para PHP entiende espacios de nombres y definiciones de tipo variable en los comentarios que mejora la finalización de código y navegación hipervínculo. (30)

Para el desarrollo de la solución fue seleccionado como IDE de desarrollo NetBeans en su versión 6.9.

2.8 Sistema Gestor de Base de Datos

Un Sistema Gestor de Base de Datos (SGBD) es una colección de programas cuyo objetivo es servir de interfaz entre la base de datos (BD), el usuario y las aplicaciones. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. Un SGBD permite definir los datos a distintos niveles de abstracción y manipular dichos datos, garantizando la seguridad e integridad de los mismos. (31)

MySQL y PostgreSQL son los principales SGBD que soporta el CMS Drupal.

2.8.1 MySQL

MySQL es un sistema de administración para BD relacionales que provee una solución robusta a los usuarios con poderosas herramientas multiusuario, utiliza soluciones SQL (del inglés Structured Query Language, Lenguaje de Consulta Estructurado). Es rápido, robusto y fácil de utilizar.

Entre las principales características podemos destacar:

- ✓ Aprovecha la potencia de sistemas multiprocesadores, gracias a su implementación multihilos.
- ✓ Soporta gran cantidad de tipos de datos para las columnas.
- ✓ Gran portabilidad entre sistemas.
- ✓ Soporta hasta 32 índices por tabla.
- ✓ Gestión de usuarios y contraseñas, manteniendo un muy buen nivel de seguridad en los datos.

2.8.2 PostgreSQL

PostgreSQL es un sistema de gestión de BD relacional, distribuido bajo la licencia BSD y con su código fuente disponible libremente. Además, es orientado a objetos, por lo que brinda elementos como herencia, tipos de datos, funciones, restricciones, disparadores y procedimientos almacenados. Posee lenguaje procedimental de alta disponibilidad y recuperación de fallas.

Este gestor tiene numerosas características que lo diferencian de otros gestores, a continuación se presentan algunas de ellas:

- ✓ Es compatible con conjuntos de caracteres internacionales, las codificaciones de caracteres *multibyte*, *unicode*, y las configuraciones regionales para la ordenación, mayúsculas y minúsculas, y el formato.
- ✓ Es altamente escalable, tanto en la gran cantidad de datos que puede manejar como en el número de usuarios concurrentes que puede acomodar.
- ✓ Ahorros considerables en costos de operación ya que ha sido diseñado y creado para tener un mantenimiento y ajuste mucho menor que otros productos, conservando todas las características, estabilidad y rendimiento.
- ✓ Multiplataforma. (32)

El uso de este gestor de BD trae consigo una serie de beneficios y ventajas para el usuario, a continuación se especifican algunas de estas:

- ✓ Es altamente extensible, soporta operadores y tipos de datos definidos por el usuario.
- ✓ Posee documentación muy bien organizada, pública y libre, con comentarios de los propios usuarios, facilitando de esta forma el trabajo del usuario con la BD.

- ✓ Existen comunidades muy activas, varias de ellas en castellano, lo cual facilita la investigación del gestor.
- ✓ Contiene utilidades para limpieza de la BD, permitiendo de esta forma evacuar los datos en otro lugar y volver a tener la BD limpia para los datos más actuales.
- ✓ Incluye utilidades para análisis y optimización de consultas, haciendo así más eficiente el trabajo de los desarrolladores al facilitar los procesos de obtención de datos. (33)

Además de las ventajas mencionadas, PostgreSQL no tiene límites para la cantidad de datos que se puede guardar en una tabla. (34) Elemento que es muy conveniente para el sistema que se desea desarrollar, ya que no se puede saber con exactitud qué cantidad de datos pueden arrojar las pruebas de seguridad y por ende no es posible saber cuánta información se necesita guardar.

2.9 Herramienta de desarrollo para el proceso de integración de datos ETL

La plataforma Pentaho de código abierto para la inteligencia de negocio cubre amplias necesidades para realizar el análisis de datos de gran volumen y variedad. Esta plataforma es considerada como la construcción del futuro de la analítica de negocios y está compuesta por una gran variedad de herramientas, pero teniendo en cuenta las necesidades de la investigación solo se tuvo en cuenta Pentaho Data Integration (PDI). PDI es una herramienta de integración de datos, la cual incluye un conjunto de herramientas para realizar este proceso ETL. Uno de sus objetivos es que la integración sea fácil de generar, mantener y desplegar. PDI abre, limpia e integra la información y la pone en manos del usuario.

Esta herramienta se caracteriza por ser de código libre y sin costes de licencia, y además:

- ✓ Ofrece un entorno gráfico de desarrollo.
- ✓ Uso de tecnologías estándar: Java, XML, JavaScript.
- ✓ Fácil de instalar y configurar.
- ✓ Multiplataforma.

PDI incluye las siguientes herramientas:

- ✓ **Spoon:** es la herramienta gráfica que permite el diseño de las transformaciones y trabajos. Incluye opciones para previsualizar y probar los elementos desarrollados. Es la principal herramienta de trabajo de PDI y con la que se construyen y validan los procesos ETL.
- ✓ **Pan:** es la herramienta que permite la ejecución de las transformaciones diseñadas en Spoon (bien desde un fichero o desde el repositorio). Permite desde la línea de comandos preparar la ejecución mediante *scripts*.
- ✓ **Kitchen:** similar a la herramienta Pan, pero se usa para ejecutar los trabajos. (35)

Las ventajas que propician estas herramientas para facilitar el trabajo con la información, serán utilizadas para llevar a cabo la integración de los datos de forma eficiente, con el objetivo de crear un proceso de evaluación con calidad. Se utilizarán para integrar los datos obtenidos por las pruebas de seguridad realizadas mediante la herramienta W3AF a un *software* con el Sistema de evaluación de seguridad, para garantizar que el proceso se realice en el menor tiempo posible y con mayor eficiencia.

2.10 Conclusiones del capítulo

Con el estudio de las herramientas y tecnologías existentes fue posible realizar una adecuada selección de las mismas, para dar solución a la problemática planteada. Además, su análisis permitió conocer sus características, ventajas y desventajas haciendo posible que los resultados obtenidos con su posterior aplicación sean lo más factible y eficiente posible.

CAPÍTULO 3

CARACTERÍSTICAS Y DISEÑO DEL SISTEMA

3. Introducción

El propósito fundamental del presente capítulo es identificar las actividades que se deben realizar según las fases o etapas definidas en la metodología de desarrollo. Se exponen además las características de la herramienta que se desea desarrollar para dar solución a la problemática existente. Se efectúa el levantamiento y descripción de los escenarios, así como la selección de los requisitos de calidad del servicio adecuados al sistema que se desea desarrollar. Finalizando el capítulo con elementos del diseño, necesarios para dar cumplimiento a las funcionalidades identificadas.

3.1 Fase Visión y Alcance

En todo proceso de desarrollo es indispensable tener presente la visión y el alcance del producto a desarrollar para mantenerlo enfocado a las necesidades que va a resolver. En el presente epígrafe se documenta la fase Visión y Alcance, proporcionándole al equipo de trabajo una idea de cuál sería la manera más óptima para dar solución al problema planteado.

3.1.1 Propuesta de solución

Teniendo en cuenta la problemática a resolver se propone el desarrollo de un sistema para la evaluación de la seguridad, que permita procesar los datos arrojados por las pruebas de seguridad realizadas a un *software* utilizando la herramienta W3AF. La solución contará con diferentes funcionalidades las cuales le permitirán al usuario almacenar los datos del proyecto a evaluar, para posteriormente ser procesados mediante ETL, obteniendo de ellos solo la información necesaria para llevar a cabo la evaluación del *software* mediante el uso de métricas de seguridad. Además, brindará la opción de generar un reporte con el objetivo de brindar la información necesaria al usuario sobre el nivel de seguridad de su sistema.

Para el proceso ETL dentro de la propuesta de solución se utilizaron las siguientes reglas para garantizar la rapidez y calidad de las transacciones en la BD:

- ✓ **Entrada Fichero de Texto**: regla que se utiliza para obtener los datos del fichero con las pruebas de seguridad. Mediante una expresión regular se carga el archivo con los datos y se divide el archivo en varias columnas.
- ✓ **Eliminar espacios en blancos**: en la columna que se guardan las vulnerabilidades encontradas se eliminan los espacios en blanco para garantizar la limpieza de estos datos.
- ✓ **Si existen valores nulos**: regla que se encarga de verificar si existen campos vacíos y de ser así los completa con un valor por defecto.
- ✓ **Cortar Cadena**: es utilizada para obtener de un campo específico únicamente la cantidad de caracteres necesarios.
- ✓ **Guardar en BD**: regla para guardar en la BD correspondiente los resultados de las reglas anteriores.

3.1.2 Vista conceptual del sistema

La guía para el desarrollo de los elementos que formarán parte de la propuesta de solución expuesta en el epígrafe anterior, se basa en las investigaciones previas desarrolladas por los autores debido a que el sistema surge por la necesidad del Departamento Seguridad Digital. Por tanto, los procesos de negocio no están bien estructurados, por lo que se presenta una vista conceptual de forma general de la aplicación para capturar los objetos más importantes, los eventos que suceden en el entorno donde estará el sistema y la relación existente entre ellos.

Descripción:

- ✓ **Evaluador:** representa la persona que interactúa con el sistema.
- ✓ **Sistema_Evaluación:** representa la aplicación web con la cual el evaluador va a interactuar para realizar el proceso de evaluación.
- ✓ **Gestionar_Proyectos:** funcionalidad del sistema encargada de gestionar los datos del proyecto que será evaluado.

- ✓ **Realizar_Evaluación:** funcionalidad del sistema encargada de almacenar en el servidor los datos correspondientes a las pruebas de seguridad y posteriormente iniciar el proceso ETL para estos datos.
- ✓ **Generar_Reporte:** funcionalidad que se encarga de mostrar al evaluador un reporte con el nivel de seguridad de su aplicación una vez finalizado el proceso de evaluación.
- ✓ **BD_Evaluación:** representa la BD encargada de registrar todo el proceso de extracción, transformación y carga de los datos.

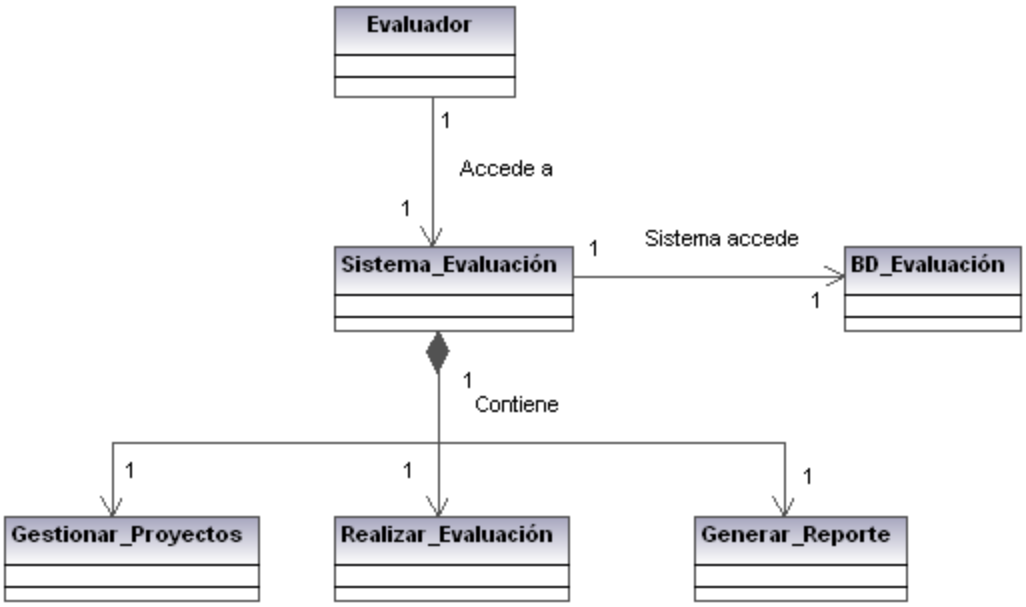


Figura 1: Modelo conceptual del sistema

3.1.3 Definición de personas

Las personas representan individuos o sistemas externos que van a interactuar con la aplicación.

Personas	Descripción
Evaluador	Encargado de:

	1. Gestionar los datos del proyecto a evaluar. 2. Subir al sistema los datos necesarios para realizar el proceso de evaluación.

Tabla 3: Definición de personas

3.2 Fase Planificación

Es esta fase se determina la planificación del proyecto, el equipo prepara las especificaciones funcionales, se realiza el proceso de diseño de la solución y los planes de trabajo. Los principales artefactos de esta fase son los escenarios y los requisitos de calidad del servicio, los cuales son utilizados para especificar los requisitos del *software* que sirven de guía para todo el proceso de desarrollo. En esta fase además serán descritos elementos necesarios para el diseño del sistema.

3.2.1 Escenarios del sistema

Un escenario es un medio por el cual se define una interacción entre una persona y el sistema para lograr cumplir una meta específica. En la medida que la persona trata de alcanzar una meta, el escenario registra los pasos precisos que se toman en el intento de alcanzar ese objetivo.

En esta investigación para el desarrollo del sistema propuesto, se identificaron los escenarios siguientes:

- ✓ **SC1. Autenticar usuario**
- ✓ **SC2. Gestionar proyectos**
  - ✓ SC2.1 Insertar proyecto
  - ✓ SC2.2 Modificar proyecto
  - ✓ SC2.3 Eliminar proyecto
- ✓ **SC3. Cargar fichero**
  - ✓ SC3.1 Seleccionar fichero
  - ✓ SC3.2 Subir fichero
- ✓ **SC4. Procesar datos del fichero**

- ✓ SC4.1 Extraer datos del fichero
- ✓ SC4.2 Transformar datos del fichero
- ✓ SC4.3 Cargar datos del fichero

✓ **SC5. Generar reporte**

- ✓ SC5.1 Mostrar reporte de evaluación
- ✓ SC5.2 Exportar reporte a formato pdf

3.2.2 Priorización de los escenarios

Los escenarios se priorizan en dependencia de la importancia que tienen para los usuarios y para la validez de la aplicación. El proceso de priorizar la lista de escenarios consiste en identificar los más importantes para que sean implementados con un orden de prioridad, dada la importancia de los mismos, así como su nivel de complejidad. La calificación de la prioridad de los escenarios tiene tres niveles, Bajo, Medio y Alto con 3, 4 y 5 puntos respectivamente. En la **Tabla 4** se muestran cada uno de los escenarios definidos con su prioridad asignada.

Escenarios	Prioridad
Autenticar usuario	5
Gestionar proyectos	5
Cargar fichero	5
Procesar datos del fichero	5
Gestionar reporte	4

Tabla 4: Prioridad de los escenarios

Al realizar la priorización de los escenarios, fue posible que el equipo de trabajo se percatara de que para implementar el escenario “Procesar datos del fichero”, antes habría que desarrollar los escenarios “Autenticar usuario”, “Gestionar proyectos” y “Cargar fichero”. Además de que la implementación del escenario “Gestionar reporte” no es menos importante pero va a depender totalmente de que se implementen los demás escenarios, por lo que su prioridad es Media.

3.2.3 Requisitos de calidad del servicio

Los requisitos de calidad del servicio representan las restricciones o limitaciones con las cuales debe operar el sistema que se desarrolla, refiriéndose a características del sistema tales como: rendimiento, disponibilidad, accesibilidad, utilidad y facilidad de mantenimiento. A continuación se listan los requisitos de calidad del servicio que se identificaron para el desarrollo del sistema propuesto.

Software

- ✓ Se debe utilizar la Máquina virtual de Java 6.0.

Restricciones en el diseño y la implementación

- ✓ El SGDB deberá ser PostgreSQL 8.3.5 o superior.
- ✓ Servidor web Apache 2.0.
- ✓ El sistema debe utilizar como herramienta de desarrollo Pentaho Data Integration para llevar a cabo el proceso de integración de datos ETL.

Rendimiento

- ✓ Tener un alto nivel de respuesta para el acceso a la BD.

3.2.4 Plan de iteraciones

Para realizar de la forma más correcta y adecuada la planificación del desarrollo del sistema, se hace necesario realizar una estimación del tiempo que le tomará al programador ejecutar la codificación de cada uno de los escenarios. A partir de la prioridad asignada a cada uno de los escenarios se decide cuáles de ellos serán implementados en las primeras iteraciones, las funcionalidades críticas del sistema deben ser codificadas en iteraciones tempranas del ciclo de vida del *software*.

- ✓ **Iteración #1:** Se propone codificar los escenarios que tienen una alta prioridad, debido a que son fundamentales para el correcto funcionamiento del sistema.
- ✓ **Iteración #2:** Se propone codificar el escenario encargado de crear el reporte de la evaluación cuya prioridad es media por lo expuesto anteriormente en el **epígrafe 3.2.2**.

A continuación, en la **Tabla 5** se muestra el nombre de los escenarios, su prioridad, esfuerzo en días y su distribución por iteraciones para su implementación.

No	Escenarios	Prioridad	Esfuerzo (días)	Iteración
1	Autenticar usuario	Alta	15	1
2	Gestionar proyectos	Alta	15	1
3	Cargar fichero	Alta	20	1
4	Procesar datos del fichero	Alta	40	1
5	Generar Reporte	Media	15	2

Tabla 5: Planificación de los escenarios

3.3 Descripción de los escenarios

Para el desarrollo de la aplicación se realiza la descripción de los escenarios según las necesidades especificadas. A continuación, se muestran las descripciones de los mismos.

Nombre del escenario: Gestionar proyectos		**Identificador:** SC2
Objetivo del escenario: Gestionar todos los cambios que puedan recibir los datos de los proyectos.		
Persona: Evaluador		
Iteración: 1ra	**Prioridad:** Crítico	**Complejidad:** 1
Precondiciones: ✓ El evaluador debe haberse autenticado.		
Descripción: El evaluador selecciona la opción Gestionar proyectos. El sistema muestra un formulario donde permite seleccionar las opciones Insertar, Modificar o Eliminar un proyecto. ✓ **Nota:** Las opciones de Insertar, Modificar y Eliminar se describen como tareas del escenario.		
Validaciones: ✓ Verificar que no existan campos vacíos.		

Prototipo de interfaz de usuario 	
Requisitos de calidad del servicio	-

Tabla 6: Descripción del escenario "Gestionar proyectos"

Las descripciones de los escenarios restantes pueden ser consultadas en los anexos **(Ver Anexo I)**.

3.3.1 Especificación de tareas por escenarios

Las descripciones de las tareas pertenecientes a los escenarios podrán ser consultadas en los anexos (**Ver Anexo II**).

3.4 Elementos del diseño del sistema

Durante el diseño se modela el sistema de manera que cumpla con lo establecido por los escenarios y los requisitos de calidad del servicio. El diseño permite traducir las funcionalidades definidas, en aras de construir un *software* capaz de cumplir con todas las expectativas del cliente, lo que garantiza que durante la construcción del sistema haya un bajo nivel de cambios.

A continuación se describen elementos como la arquitectura y los patrones de diseño, basados en las particularidades que propone el CMS Drupal, el cual será utilizado para dar solución al problema planteado como quedó especificado en el capítulo anterior.

3.4.1 Especificación de la arquitectura a utilizar

Patrón arquitectónico

Los patrones arquitectónicos se utilizan para sintetizar y tener un lenguaje que describa la estructura de las soluciones. Además, definen los posibles patrones de las aplicaciones y permiten evaluar arquitecturas alternativas con ventajas y desventajas conocidas ante diferentes conjuntos de requerimientos. Un patrón arquitectónico es una transformación impuesta al diseño de todo un sistema estableciendo una estructura para todos los componentes en el mismo.

Modelo Vista Controlador (MVC) es un patrón arquitectónico que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. El patrón MVC se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página. El modelo es el SGBD y el controlador es el responsable de recibir los eventos de entrada desde la vista y realizar todas las operaciones necesarias para garantizar la comunicación entre los componentes del sistema. Por tanto, el Modelo, las Vistas y los Controladores se tratan como entidades independientes donde:

- ✓ **Modelo**: es la representación específica de la información con la cual el sistema opera. La lógica de datos asegura la integridad de estos y permite derivar nuevos datos.
- ✓ **Vista:** presenta el modelo en un formato adecuado para interactuar con la interfaz de usuario.
- ✓ **Controlador:** responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista. (36)

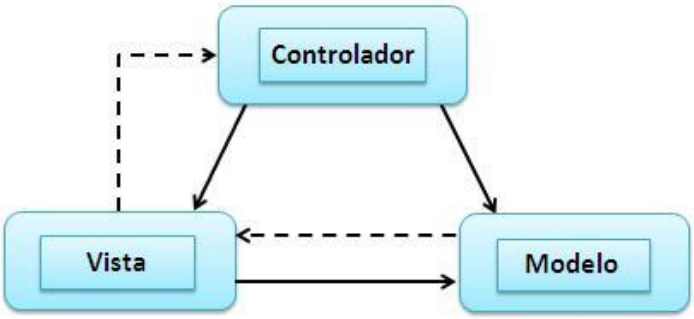


Figura 2: Patrón arquitectónico Modelo-Vista-Controlador

La arquitectura de Drupal no corresponde puramente al patrón MVC. Sin embargo, su flexible estructura separa por completo el desarrollo del aspecto a través de Temas (Vista), las interfaces para trabajar con la BD vienen en el núcleo de Drupal (Modelo) y básicamente se programan solo controladores a la hora de extender el núcleo con los nuevos módulos que se crean.

Cuando se desarrolla en Drupal ya se está desarrollando en MVC. Aunque no se esté haciendo la separación de sus archivos modelo, vista o controlador en diferentes carpetas, se están programando por separado estos aspectos. (37)

A continuación se muestra una representación de cómo se manifiesta este patrón en Drupal.

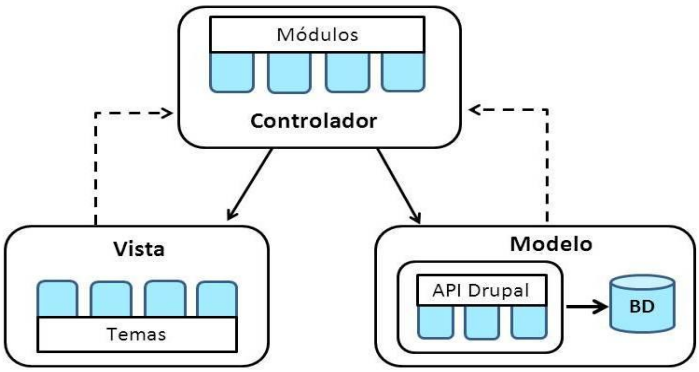


Figura 3: Patrón arquitectónico Modelo-Vista-Controlador (Drupal) (38)

Como se puede observar en la utilización que Drupal hace del patrón arquitectónico MVC no existe interacción entre la vista y el modelo, esto se debe a que dichas interacciones siempre se realizan a través del controlador.

3.4.2 Patrones de diseño

Los patrones solucionan un problema en un contexto particular. Además, impone una regla sobre la arquitectura y son usados junto a un estilo arquitectónico para determinar la forma de la estructura general de un sistema.

Patrones GoF (Gang of Four)

El funcionamiento de Drupal hace uso de algunos patrones de diseño que son propiamente de sistemas orientados a objetos, particularmente los patrones GoF. Estos se dividen en tres clasificaciones:

- ✓ **Patrones Estructurales:** describen la forma en que diferentes tipos de objetos pueden ser organizados para trabajar unos con otros.
- ✓ **Patrones de Comportamiento:** se utilizan para organizar, manejar y combinar comportamientos.
- ✓ **Patrones Creacionales:** Muestran la guía de cómo crear objetos cuando sus creaciones requieren tomar decisiones. Estas decisiones normalmente serán resueltas dinámicamente decidiendo que clases instanciar o sobre que objetos un objeto delegará responsabilidades. Su razón de ser es para facilitar, ordenar, o ayudar en la creación de objetos. (39)

Algunos de los patrones GoF que se manifiestan en el funcionamiento de Drupal son:

- ✓ **Decorator (decorador):** pertenece a la categoría de los patrones estructurales. Patrón que responde a la necesidad de añadir dinámicamente funcionalidades a un objeto. Esto permite no tener que crear sucesivas clases que hereden de la primera incorporando la nueva funcionalidad, sino otras que la implementan y se asocian a la primera. (40)

  Drupal hace amplio uso del patrón Decorator. Su uso más interesante está en el *hook_nodeapi* (), que permite a los módulos extender arbitrariamente el comportamiento de todos los nodos. Este patrón de diseño es utilizado en el Core de Drupal para facilitar la extensión de sus componentes. (41)
- ✓ **Command (comando):** pertenece a la categoría de los patrones de comportamiento. Este permite solicitar una operación a un objeto sin conocer realmente el contenido de esta operación, ni el receptor real de la misma. Para ello se encapsula la petición como un objeto, con lo que además se facilita la parametrización de los métodos. (40).

  Este patrón es utilizado en Drupal para permitir llevar a cabo la ejecución de ciertas tareas pasando como parámetro el operador, esta es la base fundamental del funcionamiento de los *hooks* (ganchos). (41)

- ✓ **Observer (Observador):** está dentro de la categoría de los patrones de comportamiento. Define una dependencia del tipo uno-a-muchos entre objetos, de manera que cuando uno de los objetos cambia su estado, el observador se encarga de notificar este cambio a todos los otros dependientes. (40)

  Al igual que en el caso del patrón de diseño Decorator, es utilizado para llevar a cabo la extensión de los componentes internos de Drupal a través de los correspondientes *hooks*. (41)
- ✓ **Singleton (instancia única):** pertenece a la categoría de los patrones creacionales. Está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Su intención consiste en garantizar que una clase únicamente tenga una instancia y proporcionar un punto de acceso global a ella. (40)

  Dentro del Core de Drupal se utiliza en diversas tareas como la gestión de las conexiones con la BD y pensando en los módulos y temas de Drupal como objetos para llevar a cabo la gestión de dichos elementos. (41)

Estos patrones le proporcionan al diseño de Drupal una gran flexibilidad y extensibilidad, posibilitando además, que su funcionamiento tenga características similares al funcionamiento de los sistemas orientados a objetos, lo cual lo convierte en una poderosa plataforma de construcción de aplicaciones web. Para conocer otros patrones de diseño remitirse al **Anexo III**.

3.4.3 Modelo de datos

Un modelo de datos es un lenguaje utilizado para describir la estructura lógica y física de la información persistente manejada por el sistema. Por lo general, permite describir los elementos que intervienen en una realidad o problema dado y la forma en que se relacionan dichos elementos entre sí. (42)

A continuación, en la **Figura 4** se muestran las entidades del modelo de datos con las que interactúa la aplicación, en este se puede observar la distribución de las tablas en la BD con sus atributos.

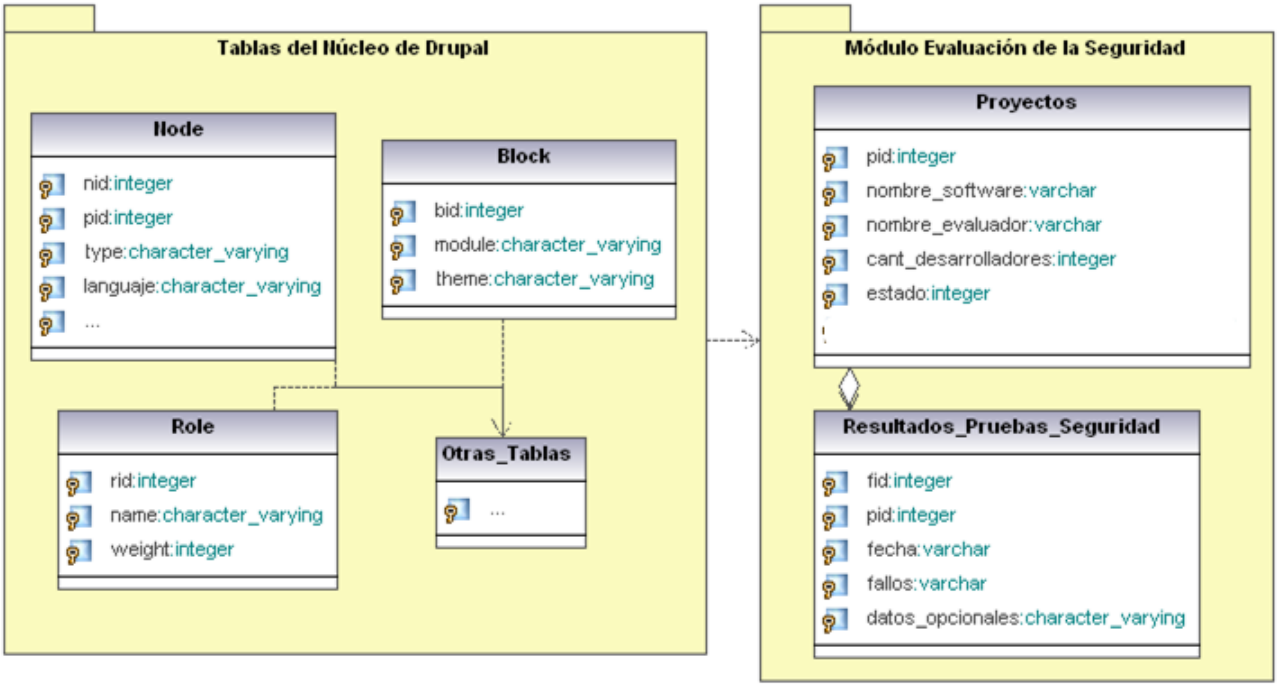


Figura 4: Modelo de datos del sistema

La imagen anterior muestra un acercamiento a la distribución de las tablas en la BD del Sistema de evaluación de seguridad. En la parte izquierda se muestran algunas tablas imprescindibles para el funcionamiento básico del CMS y a la derecha se muestran las tablas del módulo de evaluación de seguridad con sus atributos.

3.4.3.1 Descripción de las tablas

Nombre	Proyectos	
Descripción	Contiene los datos correspondientes a los proyectos que serán evaluados.	
Atributo	**Tipo**	**Descripción**
pid	integer	Identificador del proyecto.
nombre_*software*	varchar	Nombre del *software* a evaluar.

cant_desarrolladores	integer	Cantidad de desarrolladores del *software* a evaluar.
nombre_evaluador	varchar	Nombre de la persona que realiza la evaluación.
estado	integer	Estado en que se encuentra el proyecto (Activo o Inactivo).

Tabla 7: Descripción de la tabla "Proyectos"

Nombre	Resultados_Pruebas_Seguridad	
Descripción	Contiene los datos arrojados por las pruebas de seguridad, los cuales serán procesados para dar la evaluación del *software*.	
Atributo	**Tipo**	**Descripción**
fid	integer	Identificador de los resultados de las pruebas de seguridad.
pid	integer	Identificador del proyecto.
fecha	varchar	Fecha en que se realizan las pruebas de seguridad.
fallos	varchar	Vulnerabilidades encontradas mediante las pruebas de seguridad.
datos_opcionales	varchar	Otros datos arrojados por las pruebas de seguridad.

Tabla 8: Descripción de la tabla "Resultados_Pruebas_Seguridad"

3.5 Conclusiones del capítulo

La metodología seleccionada posibilitó guiar el proceso de desarrollo del sistema. La definición de las características y los escenarios del sistema, permitieron tener una visión general del mismo y realizar una planificación detallada de cada una de las fases en correspondencia con el tiempo estimado y los elementos de trabajo necesarios. La especificación de los escenarios permitió conocer al detalle la descripción de cada una de las funcionalidades, haciendo esto posible que sea más entendible su programación. La elaboración del modelo de datos hizo posible tener una visión general de la estructura de la BD del sistema, y el uso de los patrones de diseño permitió definir la estructura general del sistema a desarrollar. Con este capítulo se crearon las bases necesarias para la posterior implementación del sistema.

CAPÍTULO 4

DESARROLLO Y ESTABILIZACIÓN DEL SISTEMA

4 Introducción

En el presente capítulo se describe la solución propuesta a través de los estándares de codificación y el diagrama de despliegue, el cual contribuirá a una mejor comprensión del funcionamiento del sistema propuesto. Se describen además, las interfaces del sistema y para verificar el correcto funcionamiento de la solución se realizan un conjunto de pruebas a las principales funcionalidades.

4.1 Fase de Desarrollo

Esta es la fase donde se implementa el sistema en términos de componentes: ejecutables, ficheros de código fuente, *scripts*, entre otros. Tiene como objetivo principal desarrollar la arquitectura y el sistema como un todo, así como definir la organización del código.

4.1.1 Estándares de codificación

Los estándares de codificación consisten en estilos de codificación a la hora de escribir el código. Estos se definen teniendo en cuenta el estilo personal del programador, las características propias del lenguaje de programación, los recursos que se utilizarán y el tipo de programa que se debe implementar.

El cumplimiento de estándares hace que todo el código lleve el sello personal del programador y en caso de ser varios los programadores se busca que el código parezca que ha sido implementado por la misma persona. De esta manera, se consigue que la aplicación se convierta en un sistema fácil de comprender y de mantener.

Con el objetivo de facilitar el mantenimiento de la aplicación se emplearon los siguientes estándares de codificación establecidos por el CMS Drupal:

✓ **Funciones y variables**

Las funciones y variables deben ser nombradas usando minúsculas, y las palabras deben ser separadas utilizando un guión bajo. Las funciones deben tener además el nombre del grupo/módulo como prefijo, para evitar el conflicto de nombres entre los módulos.

✓ **Variables persistentes**

Las variables persistentes (variables/configuraciones definidas usando las funciones de Drupal *variable_get()/variable_set())* deben ser nombradas usando minúsculas y las palabras deben ser separadas utilizando un guión bajo. Deben usar el nombre del grupo/módulo como prefijo, para evitar el conflicto de nombres entre los módulos.

✓ **Operadores**

Todos los operadores binarios (operadores que están entre dos valores), como **+, -, +=, !=, ==,>**, entre otros, deben tener un espacio antes y después del operador para facilitar la lectura. Por ejemplo, una asignación debe tener el formato **$var = $foo;** en vez de **$var=$foo.** Los operadores unarios (operadores que operan sobre un solo valor), como **++**, no deben tener espacios entre el operador y la variable o número que se utiliza.

✓ **Punto y coma**

El lenguaje PHP requiere puntos y comas al final de la mayoría de las líneas, pero permite ser omitidos al final de bloques de código. Los estándares de programación de Drupal los requieren, incluso al final de bloques de código. En particular para una línea de bloques PHP.

```
<?php
print $tax;
?>
```

✓ **Inclusión de Código**

En cualquier parte que se esté incluyendo incondicionalmente un archivo de una clase, se usa *required_once*. En cualquier parte donde se esté condicionalmente incluyendo un archivo de clase (por ejemplo, métodos de fábrica) se usa *include_once*. Cualquiera de estas asegurará que el archivo sea incluido únicamente una vez. (43)

4.1.2 Diagrama de despliegue

Un diagrama de despliegue muestra las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. Este diagrama describe la arquitectura física del sistema durante la ejecución en términos de procesadores, dispositivos y componentes de *software*. (44)

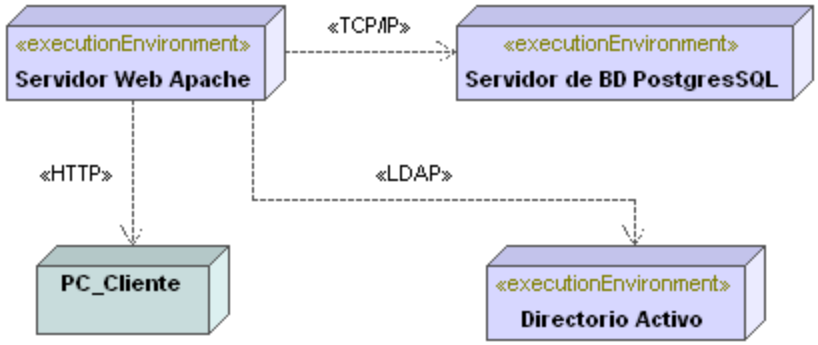


Figura 5: Diagrama de despliegue

El diagrama anterior muestra la distribución física de las relaciones entre los diferentes nodos; la cual está compuesta por un Servidor web Apache que se comunicará con el Servidor de BD PostgreSQL mediante el protocolo TCP/IP para realizar las operaciones sobre los datos del sistema. El mismo Servidor web se comunica mediante el protocolo LDAP con el Directorio Activo de la Universidad para realizar la autenticación de los usuarios y finalmente el evaluador se puede conectar al Servidor web para realizar sus peticiones mediante el protocolo HTTPS.

4.1.3 Interfaz gráfica

- ✓ Interfaz "Resultado de la evaluación": en esta interfaz es donde se muestra al evaluador el resultado final de la evaluación realizada.

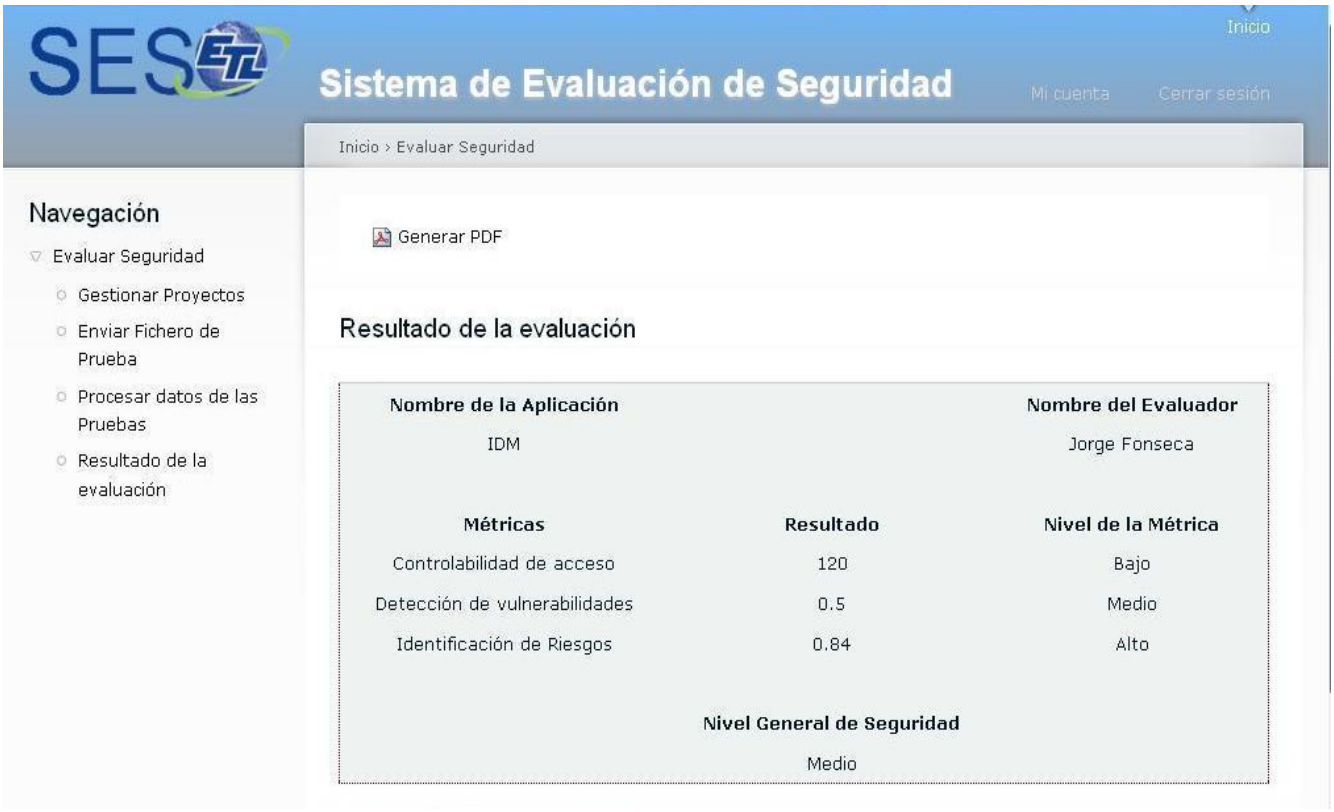


Figura 6: Interfaz "Resultado de la evaluación"

El resto de las interfaces pueden ser observadas en el **Anexo IV.**

4.2 Fase de Estabilización

Dentro de cada una de las etapas de desarrollo de un *software* las pruebas son fundamentales, ya que a partir de ellas es posible controlar que los productos cumplan requisitos mínimos de operatividad, y se puede garantizar la calidad de estos. Las pruebas constituyen una actividad en la cual un sistema es ejecutado bajo condiciones específicas, los resultados son observados y registrados a fin de determinar

si los errores ocurren cuando no tendrían que ocurrir, realizando a partir de aquí una evaluación del estado del sistema.

El primer paso que la metodología identifica para realizar las pruebas al sistema, es definir los tipos de pruebas que se van a desarrollar. Para la solución se propone realizar las pruebas unitarias mediante la técnica de caja blanca y para probar las funcionalidades de la aplicación la técnica de caja negra.

4.2.1 Pruebas unitarias. Aplicación de pruebas de caja blanca

Las pruebas unitarias son una forma de probar el correcto funcionamiento de un módulo de código. Esto se utiliza para asegurar que cada uno de los módulos funcione correctamente por separado. El objetivo de estas pruebas es el aislamiento de partes del código y la demostración de que no contienen errores, asegurándole al programador que su solución no presenta errores lógicos de programación y ante una entrada de datos determinada por el probador los valores obtenidos son los esperados.

Para aplicar las pruebas unitarias se utilizará la técnica de prueba de caja blanca la cual comprueba los caminos lógicos del *software* proponiendo casos de prueba que ejerciten conjuntos específicos de condiciones y/o ciclos. Con estas pruebas se puede examinar el estado del programa en varios puntos para determinar si el mismo coincide con el esperado. (45)

Para realizar estas pruebas se utilizó PHPUnit, conocida herramienta que constituye uno de los principales *framework* para realizar pruebas unitarias en PHP. Este *framework* permite crear y ejecutar juegos de *tests* unitarios de manera sencilla. Brinda además la posibilidad de realizar las pruebas pertinentes al código del sistema implementado, verificando que el funcionamiento de las clases sea el deseado, encontrando errores que una vez solucionados mejorarán la calidad de la aplicación. (46)

A continuación se muestra un ejemplo del resultado de las pruebas sobre las funciones *modulo_evaluacion_get_proyecto* y *cargar_datos_bd.* Estas pruebas se ejecutan sobre un archivo llamado *evaluacion.pages.Test.php* en el cual se llama a las funciones para comprobar su correcto funcionamiento. Cada punto (.) representa el resultado de una prueba.

```
3   // PHPUnit:
4   require_once('PHPUnit/Framework.php');
5
6
7   class evaluacion.module.Test extends PHPUnit_Framework_TestCase(
8
9       public function test_modulo_evaluacion_get_proyecto()
10      {
11          $modulo_evaluacion = new Modulo_Evaluacion();
12          $this->assertEquals(2, $modulo_evaluacion->modulo_evaluacion_get_proyecto(2));
13      }
14
15
16      public function test_cargar_datos_bd()
17      {
18          $modulo_evaluacion = new Modulo_Evaluacion();
19          $entry = array ();
20          $this->assertEquals($entry, $modulo_evaluacion->cargar_datos_bd($entry));
21      }
22
23  }
```

Figura 7: Prueba unitaria a la función "modulo_evaluacion_get_proyecto" y "cargar_datos_bd"

Prueba de unidad		
Nombre de la prueba: *test*_modulo_evaluacion_get_proyecto		
Estado: Satisfactoria	**Tipo:** Caja Blanca	**Ultima ejecución:** 04/05/2012
Ejecutado por: Yanisleydis Romero	**Verificado por:** Yayneris Zambrana	
Descripción: Para ejecutar la prueba se introduce de forma directa a la función el $*id* de un proyecto. Esta función se utiliza para verificar que el usuario no inserte el mismo proyecto más de una vez, y que exista solo un proyecto activo.		
Entrada: $pid		
Criterio de Aceptación: Se muestra un mensaje informando al usuario que el proyecto que intenta insertar ya existe, o que solo puede existir un proyecto activo.		

Resultado

```
PHPUnit 3.6.10 by Sebastian Bergmann.

.

Time: 0 seconds, Memory: 5.00Mb

OK (1 test, 1 assertion)
```

Tabla 9: Descripción de la prueba unitaria a la función "modulo_evaluacion_get_proyecto"

Prueba de unidad		
Nombre de la prueba: *test*_cargar_datos_bd		
Estado: Satisfactoria	**Tipo:** Caja Blanca	**Ultima ejecución:** 04/05/2012
Ejecutado por: Yanisleydis Romero	**Verificado por:** Yayneris Zambrana	
Descripción: Para ejecutar esta prueba es necesario introducir a la función un arreglo vacío en el cual serán almacenados los datos de los proyectos. El objetivo de esta función es mostrar al usuario los proyectos almacenados en la BD con sus respectivos valores.		
Entrada: $*entry*= *array*()		
Criterio de Aceptación: Se muestran los datos de los proyectos almacenados en la BD.		

Resultado

```
.

Time: 0 seconds, Memory: 5.50Mb

OK (1 test, 1 assertion)
```

Tabla 10: Descripción de la prueba unitaria a la función "cargar_datos_bd"

Las pruebas unitarias que se aplicaron a los restantes métodos para verificar su correcto funcionamiento, se pueden consultar en el **Anexo V**.

4.2.2 Aplicación de pruebas de caja negra

Las pruebas de caja negra se refieren a las pruebas que se llevan a cabo sobre la interfaz del *software*. O sea, los casos de pruebas pretenden demostrar que las funciones del producto son operativas, que la entrada se realiza de manera adecuada, que se produce un resultado correcto y que la integridad de la información externa se mantiene. (45) Las pruebas de caja negra examinan algunos aspectos del modelo fundamental del sistema sin valorar demasiado la estructura lógica interna del *software*. Se centran principalmente en los requisitos funcionales del *software* y permiten obtener un conjunto de condiciones de entrada que ejerciten completamente estos requisitos ignorándose la estructura de control.

Escenario: "Gestionar proyecto".

Tareas del Escenario	Descripción	Texto	Respuesta del Sistema	Flujo Central
Tarea: Insertar proyecto	Se procede a insertar un proyecto una vez seleccionada la opción gestionar proyecto.	Nombre *Software* Cantidad de desarrolladores Nombre Evaluador Estado (Válidos)	El sistema inserta el proyecto en la BD.	Opción Gestionar Proyectos.
		Nombre *Software* Cantidad de desarrolladores Nombre Evaluador Estado	El sistema muestra un mensaje de error informando que se entraron datos incorrectos, o campos vacíos	

		(Inválido)	que imposibilitaron la inserción.	
Tarea: Modificar proyecto	Se modifican los datos del proyecto.	Nombre *Software* Cantidad de desarrolladores Nombre Evaluador Estado (Válidos)	El sistema modifica los datos del proyecto en la BD.	Opción Gestionar Proyecto / Opción Editar.
		Nombre *Software* Cantidad de desarrolladores Nombre Evaluador Estado (Inválido)	El sistema muestra un mensaje de error informando que se entraron datos incorrectos, o campos vacíos que imposibilitaron la inserción.	
Tarea: Eliminar proyecto	Se elimina un proyecto existente.	Identificador del proyecto en la BD.	El sistema elimina de la BD el proyecto.	Opción Gestionar Proyecto / Opción Eliminar.

Tabla 11: Descripción del caso de prueba a las tareas del escenario "Gestionar proyecto"

Las descripciones de los casos de pruebas pertenecientes a los restantes escenarios pueden ser observadas en el **Anexo VI**.

4.2.3 Resultado de las pruebas

Buscar los elementos más relevantes de las pruebas, es el mayor peso a la hora de analizar los resultados una vez ejecutadas. Después de aplicar dos iteraciones de pruebas a la aplicación, de manera general las no conformidades (NC) generadas han sido solucionados en su totalidad y por tanto no afectan el desarrollo de la aplicación, por lo que se encuentra lista para ser desplegada.

En la **Figura 8** se encuentran representadas las iteraciones de pruebas realizadas, así como el número de NC generadas y corregidas en cada una de ellas.

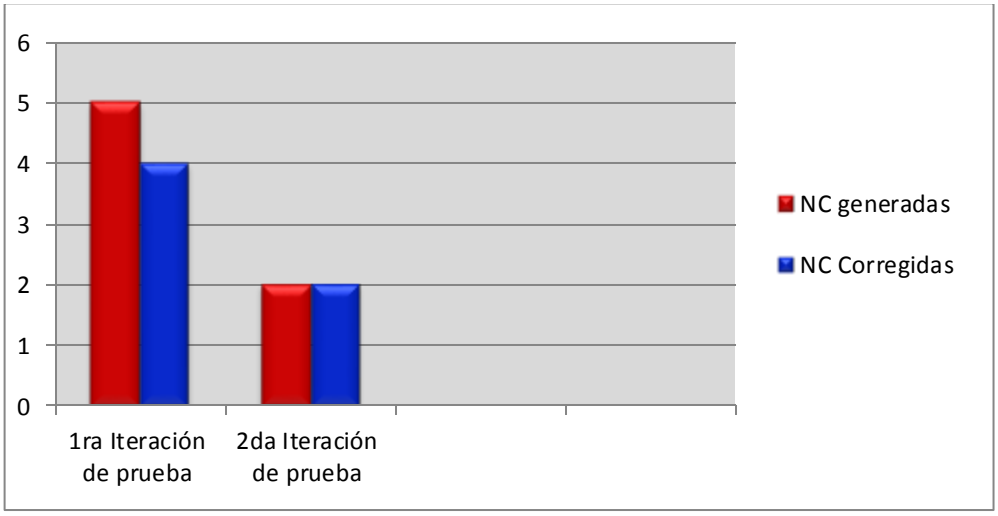


Figura 8: Iteraciones de pruebas

4.3 Conclusiones del capítulo

Definir y aplicar los estándares de codificación permitió realizar una programación ordenada y legible. Con la elaboración de los diferentes diagramas representados en el capítulo es posible tener una visión detallada de los componentes del sistema. Las pruebas realizadas contribuyeron a identificar las no conformidades y vulnerabilidades en el código, haciendo posible que el sistema cuente con una mayor calidad.

CONCLUSIONES GENERALES

Con el desarrollo del presente trabajo se logró implementar una aplicación web que permite la evaluación de la seguridad de un *software*, como resultado de la necesidad de conocer el nivel de seguridad con que cuentan las aplicaciones desarrolladas en el Departamento de Seguridad Digital del CISED. De esta manera, es posible concluir que:

- ✓ El análisis de los referentes teóricos del tema hizo posible la definición de las métricas para evaluar la seguridad en las aplicaciones y permitió realizar una correcta selección de la metodología, las tecnologías y herramientas para el desarrollo del sistema para la evaluación de la seguridad.
- ✓ Con la utilización de las herramientas para pruebas de seguridad, la selección de métricas de seguridad y el estudio del proceso de integración de datos ETL es posible determinar el nivel de seguridad de las aplicaciones.
- ✓ Identificar y describir los escenarios, así como seleccionar la arquitectura y los patrones de diseño que se deben tener en cuenta para el desarrollo del sistema, permitió comprender las características que debía poseer la solución, facilitando en gran medida su implementación.
- ✓ El desarrollo de la aplicación contribuyó a conocer la importancia de los resultados arrojados por las pruebas de seguridad, y la necesidad de su aprovechamiento para dar criterios de importancia sobre aplicaciones desarrolladas, a fin de garantizar la calidad de las mismas.
- ✓ Probar las funcionalidades del sistema a través del diseño de casos de prueba y mediante las pruebas unitarias, permitió validar el correcto funcionamiento del sistema.

Por todo lo anterior expuesto se concluye que el objetivo propuesto para el presente trabajo ha sido cumplido satisfactoriamente, ya que la aplicación da solución a la situación problemática planteada.

RECOMENDACIONES

Una vez concluido el presente trabajo de diploma se recomienda:

- ✓ Incrementar funcionalidades al sistema que permitan evaluar otros elementos de calidad.
- ✓ Que el uso de la herramienta no quede solo en el Departamento de Seguridad Digital, sino que se extienda al CISED, y posteriormente esté disponible en otros centros productivos de la UCI.

REFERENCIAS BIBLIOGRÁFICAS

1. **Fernández, Mellado y Rodríguez, Daniel.** *TECNIMAP, EVALUACION DE LA CALIDAD Y SEGURIDAD EN PRODUCTOS SOFTWARE.* 2010.

2. **Oficina Nacional de Normalización.** *NC-ISO-IEC 9126-1.* Cuba : s.n., 2005.

3. **Pressman, Roger.** *Software Engineering.* 1992.

4. —. *Ingeniería del software. Un enfoque práctico. 4ª Edición.* 1998.

5. **López, Ana María.** INTRODUCCIÓN A LA CALIDAD DE SOFTWARE. [En línea] [Citado en: octubre de 2011]. http://www.utp.edu.co/php/revistas/ScientiaEtTechnica/docsFTP/111233326-331.pdf.

6. **Contreras Chaves, Nataly.** Scribd. [En línea] [Citado en: octubre de 2011]. http://es.scribd.com/doc/59484913/Historia-de-La-Seguridad-a.

7. **Mira, José Joaquín y Gómez, José.** *Criterio, Indicador, Estándar.*

8. *ISO/IEC 14598-1:1999 Information technology - Software product evaluation -Part 1: General overview.* 1999.

9. **Ramírez Lama, Juan José.** Scribd, [Título del documento: "Requisitos y Evaluación de Calidad de Productos de Software"]. [En línea] [Citado en: octubre 2011]. http://es.scribd.com/doc/59381960/SQuaREJuaramir-TAISw.

10. *Métricas de Seguridad, Indicadores & Cuadro de Mando.* **Corletti, Alejandro y de Alba, Carmen.** 2009.

11. **Méndez, Aurora.** Reporte de Métricas. [En línea] [Citado en: octubre 2011]. http://es.scribd.com/doc/55444162/Reporte-de-Metricas-Aurora-Mm.

12. *Métricas de Seguridad,Indicadores y Cuadro de Mando.* **Corletti, Alejandro y de Alba Muñóz, Carmen.** abril 2008.

13. Eva. [En línea] http://eva.uci.cu/file.php/92/Tema_3_SW_Seguro/Materiales/Apoyo/Fundamentos_de_seguridad_en_el_software_1.1.pdf.

14. Seguridad en Sistemas y Tecnicas de Hacking. [En línea] [Citado en: octubre de 2011]. http://thehackerway.com/2011/05/30/funcionamiento-de-w3af-introduccion-al-framework-parte-i/#more-317.

15. SYBVEN it integration. [En línea] [Citado en: octubre de 2011]. http://www.sybven.com/portales/index.php?option=com_content&view=article&id=351:etl-es&catid=124:conceptos-teoricos.

16. **Mendoza Sánchez, Ing. María A.** Informatizate.net. [En línea] [Citado en: noviembre de 2011]. http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html.

17. **Microsoft.** Tutorial: Seguimiento de elementos de trabajo. [En línea] [Citado en: noviembre de 2011]. http://msdn.microsoft.com/es-es/library/ms181269%28v=vs.80%29.aspx.

18. **Hernández Orallo, Enrique.** El Lenguaje Unificado de Modelado (UML). [En línea] [Citado en: noviembre de 2011]. http://www.disca.upv.es/enheror/pdf/ActaUML.PDF.

19. **Marroquín, Néstor.** Google books. [En línea] [Citado en: noviembre de 2011]. books.google.com.cu/books?id=tSdGxtSrlU8C&pg=PA530&lpg=PA530&dq="Las+Herramientas+CASE+son+diversas+aplicaciones+informáticas+destinadas+a+aumentar+la+productividad+en+el+desarrollo+de+software+reduciendo+el+coste+de+las+mismas+en+términos+de+tiempo+y+de+.

20. **James.** freedownloadmanager. [En línea] [Citado en: noviembre de 2011]. http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%20%28Iglesia_Anglicana%29_%5BMac_OS_X_cuenta_14717_p/.

21. Ecured. [En línea] [Citado en: noviembre de 2011]. http://www.ecured.cu/index.php/Altova_Umodel.

22. Scribd, Características de Apache. [En línea] [Citado en: noviembre de 2011]. http://es.scribd.com/doc/52208534/29/CARACTERISTICAS-Y-VENTAJAS-DEL-APACHE.

23. Edujoomla.es. [En línea] [Citado en: noviembre de 2011]. http://www.edujoomla.es/que-es-joomla.

24. Pilos.Net, Características de Drupal. [En línea] [Citado en: noviembre de 2011]. http://www.pilos.com.co/drupal/27-caracteristicas-de-drupal/.

25. Enciclopedia Libre Universal en Español. [En línea] [Citado en: noviembre de 2011]. http://enciclopedia.us.es/index.php/Enciclopedia_Libre_Universal_en_Espa%C3%B1ol.

26. Ecured. [En línea] [Citado en: noviembre de 2011]. http://www.ecured.cu/index.php/PHP#Caracter.C3.ADsticas_de_PHP.

27. desarrolloweb.com. [En línea] [Citado en: noviembre 2011]. http://www.desarrolloweb.com/articulos/182.php.

28. desarrolloweb.com. [En línea] [Citado en: noviembre 2011]. http://www.desarrolloweb.com/articulos/que-es-html.html.

29. zend. [En línea] [Citado en: noviembre de 2011]. http://www.zend.com/en/products/studio/.

30. lawebdelprogramador. *Interactive Programmers Community.* [En línea] [Citado en: noviembre de 2011]. http://www.lawebdelprogramador.com/foros/Netbeans/index1.html.

31. maestros del web. [En línea] [Citado en: noviembre de 2011]. http://www.maestrosdelweb.com/principiantes/%C2%BFque-son-las-bases-de-datos/.

32. PostgreSQL. [En línea] [Citado en : noviembre de 2011]. http://www.postgresql.org/.

33. **PostgreSQL, Equipo de desarrollo.** *Guia del Programador de PostgreSQL.* s.l. : Editado por Thomas Lockhart.

34. PostgreSQL-es. [En línea] [Citado en : noviembre de 2011]. http://www.postgresql.org.es/sobre_postgresql.

35. Pentaho Data Integration. [En línea] [Citado en : noviembre de 2011]. http://churriwifi.wordpress.com/2010/05/10/16-3-construccion-procesos-etl-utilizando-kettle-pentaho-data-integration/.

36. Introducción al patrón arquitectónico MVC. [En línea] [Citado en: febrero 2012]. http://www.eugeniabahit.com/mvc/.

37. koala-soft. [En línea] [Citado en: febrero 2012]. http://www.koala-soft.com/internet/drupal.

38. assembla. [En línea] [Citado en: febrero de 2012]. http://my-svn.assembla.com/svn/maruprojects/documentation/.

39. **Gamma, Eric, y otros, y otros.** *Design Patterns: Elements of Reusable Object Oriented Software.* 1995.

40. assembla. [En línea] [Citado en: marzo de 2012]. http://my-svn.assembla.com/svn/maruprojects/documentation/.

41. drupal.org. [En línea] [Citado en: marzo de 2012]. http://drupal.org/node/282303.

42. Definición.de. [En línea] [Citado en: marzo de 2012]. [Citado el: 10 de febrero 2012] http://definicion.de/modelo-de-datos/.

43. Coding standars. *Drupal.org.* [En línea] [Citado en: marzo de 2012]. http://drupal.org/coding-standards.

44. [En línea] [Citado en: abril de 2012].
http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html.

45. Ecured. [En línea] [Citado en: abril de 2012].
http://www.ecured.cu/index.php/Flujo_de_pruebas_de_un_software.

46. desarrolloweb.com. [En línea] [Citado en: abril de 2012]. http://www.desarrolloweb.com/de_interes/phpunit-herramienta-testeo-php-4436.html.

BIBLIOGRAFÍA CONSULTADA

1. **Fernández, Mellado y Rodríguez, Daniel.** *TECNIMAP, EVALUACION DE LA CALIDAD Y SEGURIDAD EN PRODUCTOS SOFTWARE.* 2010.

2. **Oficina Nacional de Normalización.** *NC-ISO-IEC 9126-1.* Cuba : s.n., 2005.

3. **Pressman, Roger.** *Software Engineering.* 1992.

4. —. *Ingeniería del software. Un enfoque práctico. 4ª Edición.* 1998.

5. **López, Ana María.** INTRODUCCIÓN A LA CALIDAD DE SOFTWARE. [En línea] [Citado en: octubre de 2011]. http://www.utp.edu.co/php/revistas/ScientiaEtTechnica/docsFTP/111233326-331.pdf.

6. **Contreras Chaves, Nataly.** Scribd. [En línea] [Citado en: octubre de 2011]. http://es.scribd.com/doc/59484913/Historia-de-La-Seguridad-a.

7. **Mira, José Joaquín y Gómez, José.** *Criterio, Indicador, Estándar.*

8. *ISO/IEC 14598-1:1999 Information technology - Software product evaluation -Part 1: General overview.* 1999.

9. **Ramírez Lama, Juan José.** Scribd, [Título del documento: "Requisitos y Evaluación de Calidad de Productos de Software"]. [En línea] [Citado en: octubre 2011]. http://es.scribd.com/doc/59381960/SQuaREJuaramir-TAISw.

10. *Métricas de Seguridad, Indicadores & Cuadro de Mando.* **Corletti, Alejandro y de Alba, Carmen.** 2009.

11. **Méndez, Aurora.** Reporte de Métricas. [En línea] [Citado en: octubre 2011]. http://es.scribd.com/doc/55444162/Reporte-de-Metricas-Aurora-Mm.

12. *Métricas de Seguridad,Indicadores y Cuadro de Mando.* **Corletti, Alejandro y de Alba Muñóz, Carmen.** abril 2008.

13. Eva. [En línea] http://eva.uci.cu/file.php/92/Tema_3_SW_Seguro/Materiales/Apoyo/Fundamentos_de_seguridad_en_el_software_1.1.pdf.

14. Seguridad en Sistemas y Tecnicas de Hacking. [En línea] [Citado en: octubre de 2011]. http://thehackerway.com/2011/05/30/funcionamiento-de-w3af-introduccion-al-framework-parte-i/#more-317.

15. SYBVEN it integration. [En línea] [Citado en: octubre de 2011]. http://www.sybven.com/portales/index.php?option=com_content&view=article&id=351:etl-es&catid=124:conceptos-teoricos.

16. **Mendoza Sánchez, Ing. María A.** Informatizate.net. [En línea] [Citado en: noviembre de 2011]. http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html.

17. **Microsoft.** Tutorial: Seguimiento de elementos de trabajo. [En línea] [Citado en: noviembre de 2011]. http://msdn.microsoft.com/es-es/library/ms181269%28v=vs.80%29.aspx.

18. **Hernández Orallo, Enrique.** El Lenguaje Unificado de Modelado (UML). [En línea] [Citado en: noviembre de 2011]. http://www.disca.upv.es/enheror/pdf/ActaUML.PDF.

19. **Marroquín, Néstor.** Google books. [En línea] [Citado en: noviembre de 2011]. books.google.com.cu/books?id=tSdGxtSrlU8C&pg=PA530&lpg=PA530&dq="Las+Herramientas+CASE+son+diversas+aplicaciones+informáticas+destinadas+a+aumentar+la+productividad+en+el+desarrollo+de+software+reduciendo+el+coste+de+las+mismas+en+términos+de+tiempo+y+de+.

20. **James.** freedownloadmanager. [En línea] [Citado en: noviembre de 2011]. http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%20%28Iglesia_Anglicana%29_%5BMac_OS_X_cuenta_14717_p/.

21. Ecured. [En línea] [Citado en: noviembre de 2011]. http://www.ecured.cu/index.php/Altova_Umodel.

22. Scribd, Características de Apache. [En línea] [Citado en: noviembre de 2011]. http://es.scribd.com/doc/52208534/29/CARACTERISTICAS-Y-VENTAJAS-DEL-APACHE.

23. Edujoomla.es. [En línea] [Citado en: noviembre de 2011]. http://www.edujoomla.es/que-es-joomla.

24. Pilos.Net, Características de Drupal. [En línea] [Citado en: noviembre de 2011]. http://www.pilos.com.co/drupal/27-caracteristicas-de-drupal/.

25. Enciclopedia Libre Universal en Español. [En línea] [Citado en: noviembre de 2011]. http://enciclopedia.us.es/index.php/Enciclopedia_Libre_Universal_en_Espa%C3%B1ol.

26. Ecured. [En línea] [Citado en: noviembre de 2011]. http://www.ecured.cu/index.php/PHP#Caracter.C3.ADsticas_de_PHP.

27. desarrolloweb.com. [En línea] [Citado en: noviembre 2011]. http://www.desarrolloweb.com/articulos/182.php.

28. desarrolloweb.com. [En línea] [Citado en: noviembre 2011]. http://www.desarrolloweb.com/articulos/que-es-html.html.

29. zend. [En línea] [Citado en: noviembre de 2011]. http://www.zend.com/en/products/studio/.

30. lawebdelprogramador. *Interactive Programmers Community.* [En línea] [Citado en: noviembre de 2011]. http://www.lawebdelprogramador.com/foros/Netbeans/index1.html.

31. maestros del web. [En línea] [Citado en: noviembre de 2011]. http://www.maestrosdelweb.com/principiantes/%C2%BFque-son-las-bases-de-datos/.

32. PostgreSQL. [En línea] [Citado en : noviembre de 2011]. http://www.postgresql.org/.

33. **PostgreSQL, Equipo de desarrollo.** *Guia del Programador de PostgreSQL.* s.l. : Editado por Thomas Lockhart.

34. PostgreSQL-es. [En línea] [Citado en : noviembre de 2011]. http ://www.postgresql.org.es/sobre_postgresql.

35. Pentaho Data Integration. [En línea] [Citado en : noviembre de 2011]. http://churriwifi.wordpress.com/2010/05/10/16-3-construccion-procesos-etl-utilizando-kettle-pentaho-data-integration/.

36. Introducción al patrón arquitectónico MVC. [En línea] [Citado en: febrero 2012]. http://www.eugeniabahit.com/mvc/.

37. koala-soft. [En línea] [Citado en: febrero 2012]. http://www.koala-soft.com/internet/drupal.

38. assembla. [En línea] [Citado en: febrero de 2012]. http://my-svn.assembla.com/svn/maruprojects/documentation/.

39. **Gamma, Eric.** *Design Patterns: Elements of Reusable Object Oriented Software.* 1995.

40. assembla. [En línea] [Citado en: marzo de 2012]. http://my-svn.assembla.com/svn/maruprojects/documentation/.

41. drupal.org. [En línea] [Citado en: marzo de 2012]. http://drupal.org/node/282303.

42. Definición.de. [En línea] [Citado en: marzo de 2012]. [Citado el: 10 de febrero 2012] http://definicion.de/modelo-de-datos/.

43. Coding standars. *Drupal.org.* [En línea] [Citado en: marzo de 2012]. http://drupal.org/coding-standards.

44. [En línea] [Citado en: abril de 2012]. http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html.

45. Ecured. [En línea] [Citado en: abril de 2012]. http://www.ecured.cu/index.php/Flujo_de_pruebas_de_un_software.

46. desarrolloweb.com. [En línea] [Citado en: abril de 2012]. http://www.desarrolloweb.com/de_interes/phpunit-herramienta-testeo-php-4436.html.

47. **Fernández Escribano, Gerardo.** Introduccióna Extreme Programming . [En línea] Septiembre de 2007. <http://www.info-ab.uclm.es/asignaturas/42551/trabajosAnteriores/Presentacion.

48. **Zambrana Hernandez, Yaineris.** *Medición de la calidad de Software durante el proceso de Pruebas en el Proyecto Modernización del CICPC.* Ciudad de la Habana : s.n., 2008.

49. *Visual Paradigm For Uml.* [En línea] [Citado el: 12 de Enero de 2011.] http://www.slideshare.net/vanquishdarkenigma/visual-paradigm-for-uml.

50. [En línea] [Citado el: 16 de Enero de 2011.] http://acsblog.es/articulos/trunk/LinuxActual/Apache/html/x31.html.

51. **Velandia , Miguel José.** Características de PostgreSQL. [En línea] http://es.scribd.com/miguel_velandia_1/d/69338308/36-Caracteristicas-del-PostgreSQL.

52. Introducción a HTML. [En línea] http://www.aulaclic.es/html/t_1_1.htm.

53. [En línea] [Citado el: 5 de Mayo de 2011.] http://programacionextrema.tripod.com/fases.htm.

54. desarrolloweb.com. [En línea] http://www.desarrolloweb.com/articulos/sistemas-gestores-bases-datos.html.

55. Ecured. [En línea] http://www.ecured.cu/index.php/Dise%C3%B1ador_de_pruebas.

56. PHP. [En línea] http://www.php.net/.

57. IngenieroSoftware, Patrones de diseño. [En línea] http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php.

58. definicion.de. [En línea] http://definicion.de/seguridad/.

59. *ISO/IEC 14598-5, International Standard, Information technology - Software product evaluation - Part 5: Process for evaluators.* 1998.

GLOSARIO DE TÉRMINOS

B

Base de Datos (BD): Conjunto de datos que pertenecen al mismo contexto almacenados sistemáticamente. En una BD, la información se organiza en campos y registros.

C

CISED: Centro de Identificación y Seguridad Digital.

F

***Framework*:** un *framework*, en el desarrollo de *software* es una estructura de soporte definida, en la cual otro proyecto de *software* puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros *software* para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

G

GPL: Es la Licencia Pública General de GNU, es una licencia creada por la Free *Software* Foundation en 1989 (la primera versión), y está orientada principalmente a proteger la libre distribución, modificación y uso de *software*. Su propósito es declarar que el *software* cubierto por esta licencia es *software* libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

H

HTML: Lenguaje de Marcado de Hipertexto.

HTTP: Hypertext Transfer Protocol (en español protocolo de transferencia de hipertexto) es el protocolo usado en cada transacción de la World Wide Web. Este sigue el esquema petición-respuesta entre un cliente y un servidor.

I

Indicador: Es un valor que se obtiene comparando datos lógicamente relacionados, referentes al comportamiento de una actividad, proceso o control, dentro de un tiempo específico.

L

Las fases de MSF Ágil: son grupos de actividades que llevan a cada uno de los puntos de control que guían el desarrollo, es decir, abordar las cuestiones que se refieren a gastos de tiempo y dinero ya que representan el

proceso en tiempo. Las fases no se refieren necesariamente a la ejecución de las tareas o la división del trabajo que se manejan en los ciclos. Tenga en cuenta que las fases se superponen considerablemente y que existe una retroalimentación continua entre las actividades de diferentes fases.

LDAP: El Protocolo Ligero de Acceso a Directorios es un protocolo a nivel de aplicación que permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red.

M

MSF (Microsoft Solution *Framework*): *Framework* que brinda Microsoft para guiar el desarrollo de sistemas. Es una metodología flexible e interrelacionada con una serie de conceptos, modelos y prácticas de uso, que controlan la planificación, el desarrollo y la gestión de proyectos tecnológicos.

Multiplataforma: Es un término usado para referirse a los programas, sistemas operativos, lenguajes de programación, u otra clase de *software*, que puedan funcionar en diversas plataformas. Una plataforma es una combinación de hardware y *software* usada para ejecutar aplicaciones.

S

SC: abreviatura definida para identificar los escenarios del sistema.

T

Triggers: en una BD, es un procedimiento que se ejecuta cuando se cumple una condición establecida al realizar una operación. Dependiendo de la BD, los triggers pueden ser de inserción (*INSERT*), actualización (*UPDATE*) o borrado (*DELETE*). Algunas BD pueden ejecutar triggers al crear, borrar o editar usuarios, tablas, bases de datos u otros objetos.

U

UML: Lenguaje Unificado de Modelado es el lenguaje de modelado de sistemas de *software* que se utiliza para especificar, visualizar, modificar, construir y documentar los artefactos que se obtienen durante el desarrollo.

Usuario: Persona que tiene una cuenta en una determinada computadora por medio de la cual puede acceder a los recursos y servicios que ofrece una red.

X

XML: Lenguaje de Marcado Extensible, es un metalenguaje extensible de etiquetas.

ANEXOS

Anexo I: Descripción de los escenarios

Nombre del escenario: Autenticar usuario		**Identificador:** SC1
Objetivo del escenario: Verificar que el usuario tenga acceso al sistema.		
Persona: Evaluador		
Iteración: 1ra	**Prioridad:** Crítico	**Complejidad:** 1
Precondiciones		
Descripción: El evaluador introduce su usuario y contraseña, el sistema verifica la autenticidad de los datos, si los datos son correctos el evaluador accede al sistema.		
Validaciones: ✓ Verificar que no existan campos vacíos.		
Prototipo de interfaz de usuario 		
Requisitos de calidad del servicio		

Tabla 12: Descripción del escenario "Autenticar usuario"

Nombre del escenario: Cargar fichero	**Identificador:** SC3
Objetivo del escenario: Cargar en la BD el fichero donde están almacenados los resultados arrojados	

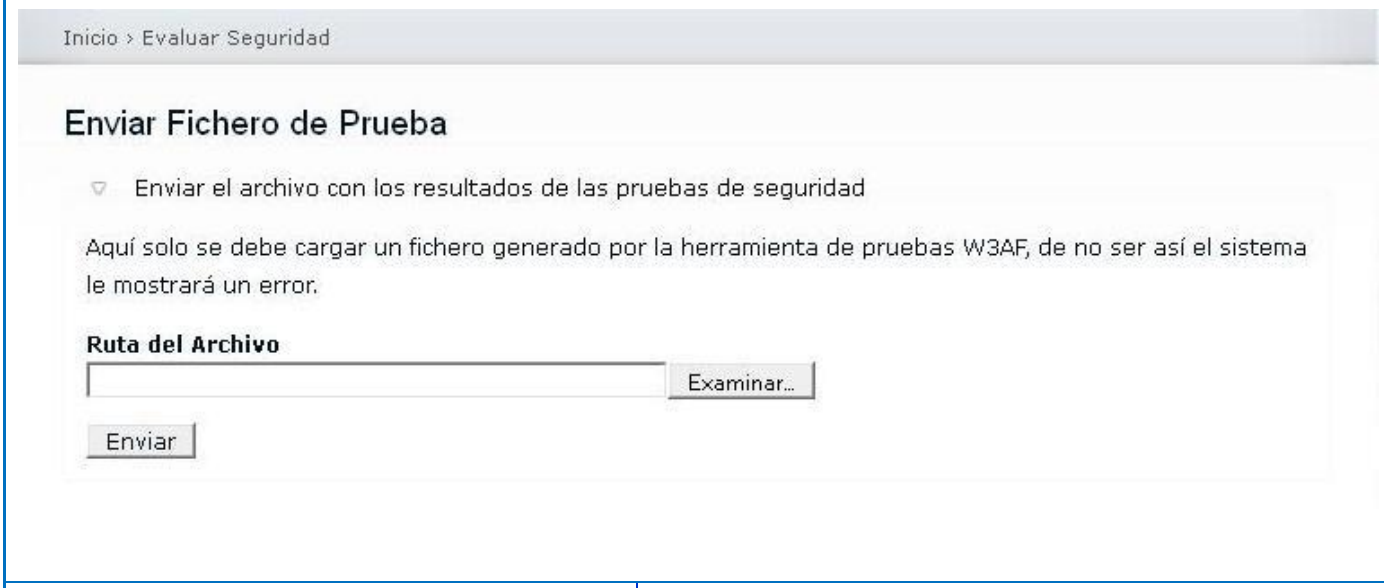
por las pruebas de seguridad.		
Persona: Evaluador		
Iteración: 1ra	**Prioridad:** Crítico	**Complejidad:** 1
Precondiciones: ✓ Debe existir un proyecto para evaluar.		
Descripción: El evaluador selecciona la opción enviar fichero de pruebas, el sistema muestra el formulario donde se encuentran los elementos necesarios para que sea cargado el fichero en la BD.		
Validaciones: ✓ Verificar que el campo no esté vacío.		
Prototipo de interfaz de usuario 		
Requisitos de calidad del servicio	-	

Tabla 13: Descripción del escenario "Cargar fichero"

Nombre del escenario: Procesar datos del fichero	**Identificador:** SC4
Objetivo del escenario: Realizar ETL a los datos del fichero cargado.	
Persona: Evaluador	

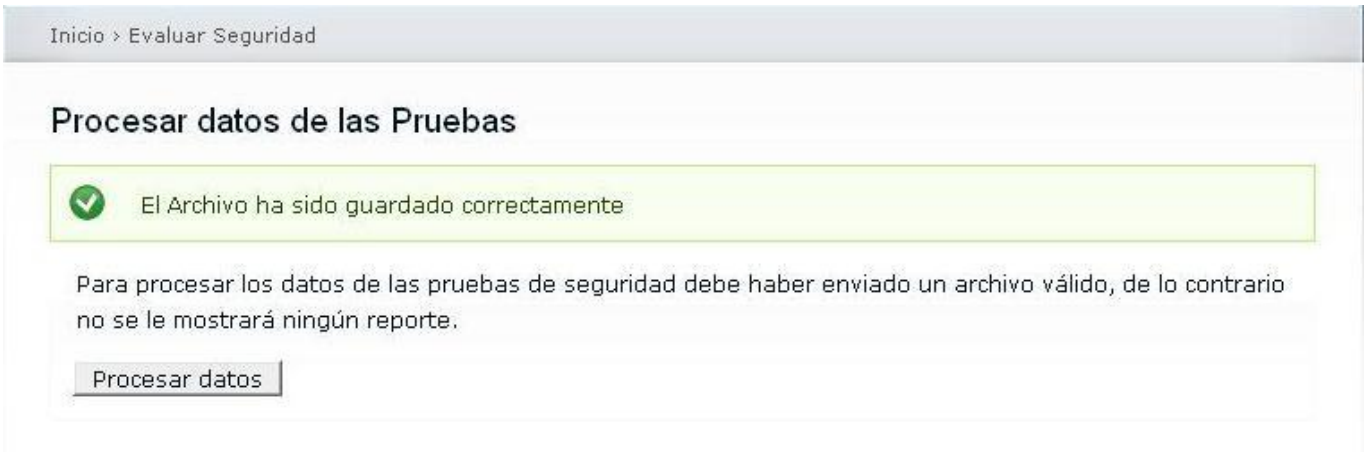
Iteración: 1ra	**Prioridad:** Crítico	**Complejidad:** 1
Precondiciones: ✓ Se debe haber cargado el fichero con los datos de las pruebas de seguridad.		
Descripción: El evaluador selecciona la opción procesar datos de las pruebas y a partir de ahí comienza el proceso ETL.		
Validaciones		
Prototipo de interfaz de usuario 		
Requisitos de calidad del servicio	✓ SGBD PostgreSQL 8.3.5 o superior ✓ Máquina virtual de Java ✓ Pentaho Data Integration ✓ Servidor web Apache	

Tabla 14: Descripción del escenario "Procesar datos del fichero"

Nombre del escenario: Generar reporte		**Identificador:** SC5
Objetivo del escenario: Permitirle al evaluador obtener los resultados de la evaluación a su aplicación.		
Persona: Evaluador		
Iteración: 2da	**Prioridad:** Secundario	**Complejidad:** 1
Precondiciones:		

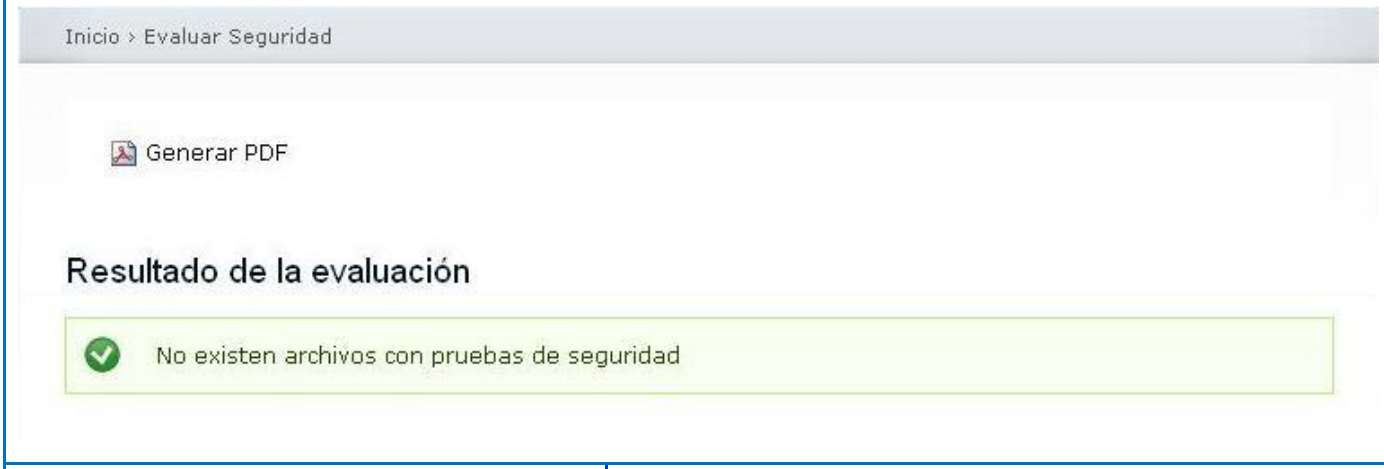
✓ Debe haberse realizado la evaluación.	
Descripción: El evaluador selecciona la opción reporte, el sistema le brinda opciones para visualizar el reporte y exportarlo como pdf.	
Validaciones	
Prototipo de interfaz de usuario 	
Requisitos de calidad del servicio	-

Tabla 15: Descripción del escenario "Generar reporte"

Anexo II: Descripción de tareas por escenario

✓ **Especificación de tareas del escenario: "Gestionar proyecto".**

El escenario se dividió en 3 tareas fundamentales, que proveerán las funcionalidades de Insertar, Modificar y Eliminar proyecto.

Nombre de la tarea: Insertar proyecto		**Identificador:** SC2.1
Objetivo de la tarea: Insertar datos del proyecto a evaluar.		
Persona: Evaluador		
Iteración: 1ra	**Prioridad:** Crítico	**Complejidad:** 1
Precondiciones		

Descripción: El evaluador accede a la aplicación y el sistema muestra un formulario con los campos necesarios a llenar, el evaluador introduce los datos del proyecto según corresponda. Los datos a introducir por el evaluador se definen a continuación:

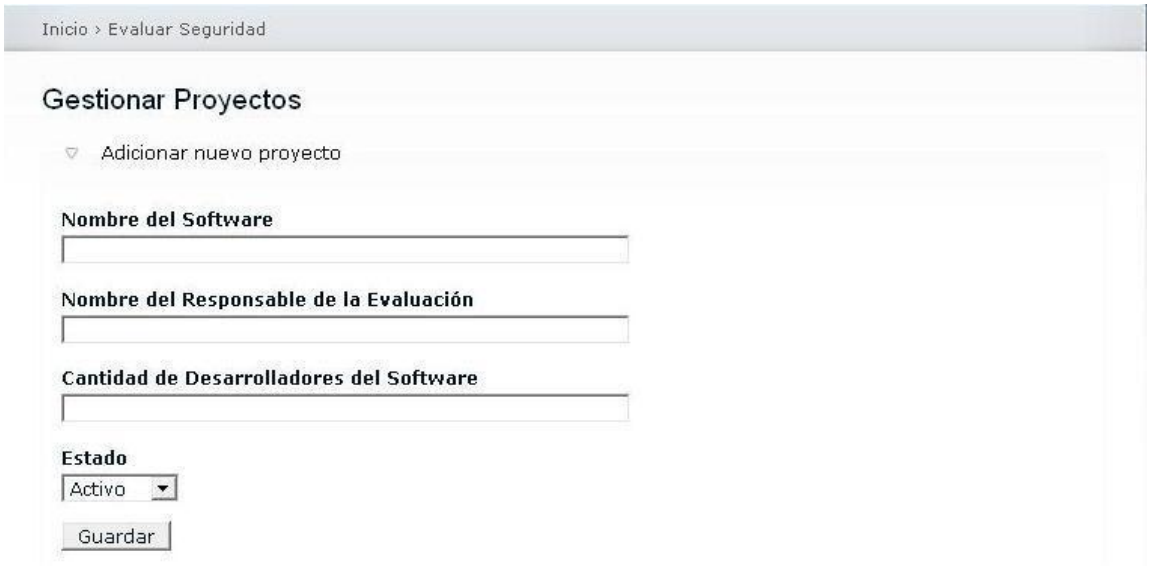
1. Nombre del *software*.
2. Nombre del evaluador.
3. Cantidad de desarrolladores del *software*.
4. Estado del proyecto.

Una vez que el evaluador inserta los datos, selecciona la opción aceptar y finalmente el sistema envía los datos a la BD.

Validaciones:

- ✓ Verificar que no existan campos vacíos.

Prototipo de interfaz de usuario



Requisitos de calidad del servicio	-

Tabla 16: Descripción de la tarea "Insertar proyecto"

Nombre de la tarea: Modificar proyecto	**Identificador:** SC2.2

Objetivo de la tarea: Modificar datos del proyecto.		
Persona: Evaluador		
Iteración: 1ra	**Prioridad:** Secundario	**Complejidad:** 1
Precondiciones		
Descripción: El evaluador accede a la opción gestionar proyectos, el sistema muestra un formulario con los proyectos disponibles, el evaluador selecciona el proyecto a modificar, modifica los datos que desee y selecciona aceptar, el sistema actualiza los datos en la BD.		
Validaciones: ✓ Verificar que no existan campos vacíos.		
Prototipo de interfaz de usuario		
		
Requisitos de calidad del servicio		

Tabla 17: Descripción de la tarea "Modificar proyecto"

Nombre de la tarea: Eliminar proyecto		**Identificador:** SC2.3
Objetivo de la tarea: Eliminar proyectos existentes.		
Persona: Evaluador		
Iteración: 1ra	**Prioridad:** Secundario	**Complejidad:** 1
Precondiciones		
Descripción: El evaluador accede a la opción gestionar proyectos, el sistema muestra los proyectos existentes, el evaluador selecciona el proyecto que desea eliminar y presiona la opción eliminar, el sistema elimina el proyecto de la BD.		
Validaciones		
Prototipo de interfaz de usuario		

Nombre	Responsable	Cantidad de Desarrolladores	Estado	Opciones
IDM	Jorge Fonseca	3	Activo	Editar|Eliminar

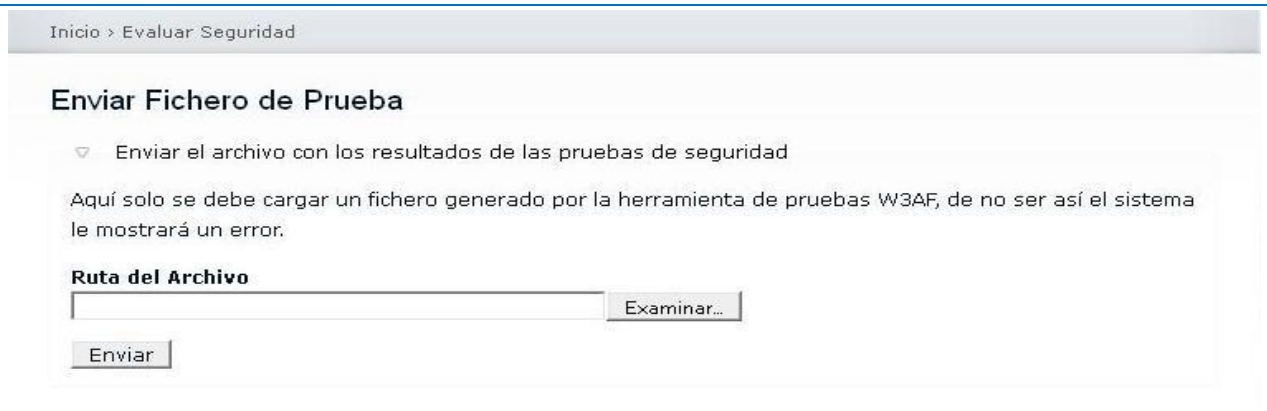
Requisitos de calidad del servicio	-

Tabla 18: Descripción de la tarea "Eliminar proyecto"

- ✓ **Especificación de tareas del escenario: "Cargar fichero".**

El escenario "Cargar fichero" se dividió en las tareas: Seleccionar fichero y Subir fichero.

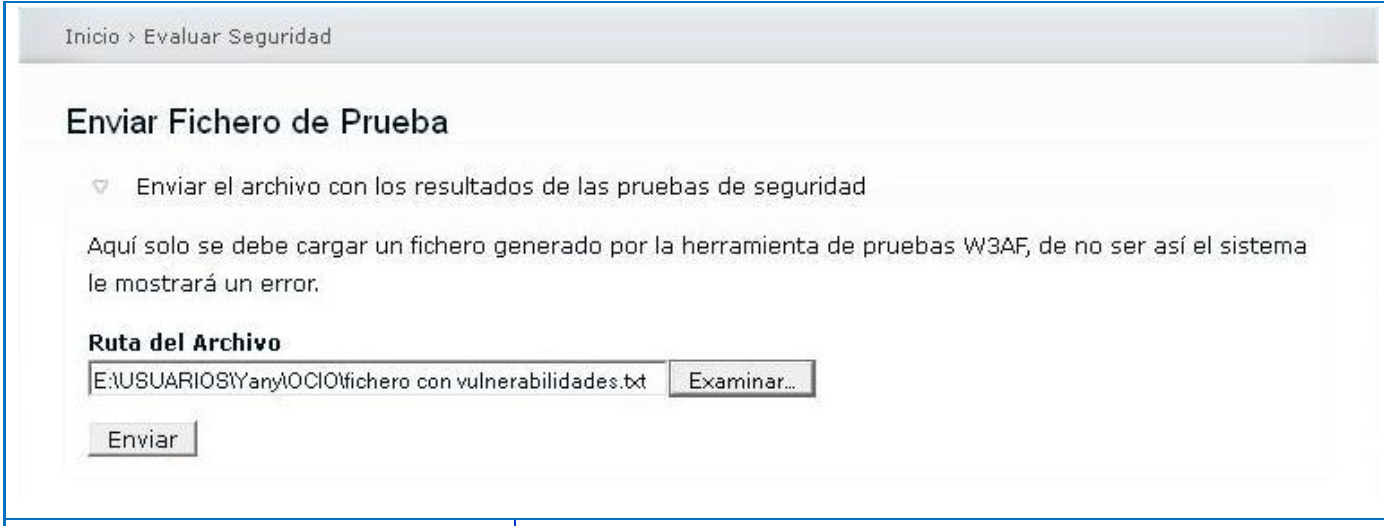
Nombre de la tarea: Seleccionar fichero		**Identificador:** SC 3.1
Objetivo de la tarea: Seleccionar el fichero donde están almacenados los resultados arrojados por las pruebas de seguridad.		
Persona: Evaluador		
Iteración: 1ra	**Prioridad:** Crítico	**Complejidad:** 1
Precondiciones: ✓ Debe existir un proyecto para evaluar.		
Descripción: El evaluador oprime el botón examinar para seleccionar el fichero arrojado por las pruebas de seguridad, selecciona el fichero a cargar, presiona el botón enviar y el sistema muestra un mensaje informando que el fichero fue cargado correctamente.		
Validaciones: ✓ Verificar que el fichero se haya cargado de forma correcta, que no existan campos vacíos.		
Prototipo de interfaz de usuario		



Requisitos de calidad del servicio	-

Tabla 19: Descripción de la tarea "Seleccionar fichero"

Nombre de la tarea: Subir fichero		**Identificador:** SC 3.2
Objetivo de la tarea: Subir el fichero seleccionado para procesar los datos.		
Persona: Evaluador		
Iteración: 1ra	**Prioridad:** Crítico	**Complejidad:** 1
Precondiciones: ✓ Debe haber seleccionado un fichero a procesar.		
Descripción: El sistema recibe el fichero cargado y lo guarda en el servidor.		
Validaciones		
Prototipo de interfaz de usuario		



Requisitos de calidad del servicio	✓ SGBD PostgreSQL 8.3.5 o superior ✓ Servidor web Apache

Tabla 20: Descripción de la tarea "Subir fichero"

- ✓ **Especificación de tareas del escenario: "Procesar datos del fichero".**

El escenario se dividió en las tareas: Extraer, Transformar y Cargar datos del fichero.

Nombre de la tarea: Extraer datos del fichero		**Identificador:** SC 4.1
Objetivo de la tarea: Extraer los datos del fichero cargado anteriormente.		
Persona		
Iteración: 1ra	**Prioridad:** Crítico	**Complejidad:** 1
Precondiciones: ✓ Se debe haber cargado el fichero con los datos de las pruebas de seguridad.		
Descripción: Una vez cargado el fichero con los datos en la BD, se procede a la extracción de los mismos. El sistema extrae los datos necesarios para realizar posteriormente las transformaciones.		
Validaciones		
Prototipo de interfaz de usuario		
Requisitos de calidad del servicio	✓ SGBD PostgreSQL 8.3.5 o superior	

	✓ Máquina virtual de Java ✓ Pentaho Data Integration ✓ Servidor web Apache

Tabla 21: Descripción de la tarea "Extraer datos del fichero"

Nombre de la tarea: Transformar datos del fichero		**Identificador:** SC 4.2
Objetivo de la tarea: Transformar los datos de la extracción.		
Persona		
Iteración: 1ra	**Prioridad:** Crítico	**Complejidad:** 1
Precondiciones: ✓ Se deben haber extraído los datos necesarios.		
Descripción: Una vez extraídos los datos, se procede a la transformación de los mismos. El sistema inicia el proceso de transformación el cual mediante la opción partir campos selecciona únicamente los datos necesarios para realizar el proceso de evaluación.		
Validaciones		
Prototipo de interfaz de usuario		
Requisitos de calidad del servicio	✓ Máquina virtual de Java ✓ Pentaho Data Integration ✓ Servidor web Apache	

Tabla 22: Descripción de la tarea "Transformar datos del fichero"

Nombre de la tarea: Cargar datos del fichero		**Identificador:** SC 4.3
Objetivo de la tarea: Cargar los datos de la transformación.		
Persona		
Iteración: 1ra	**Prioridad:** Crítico	**Complejidad:** 1
Precondiciones: ✓ Se deben haber transformado los datos necesarios.		

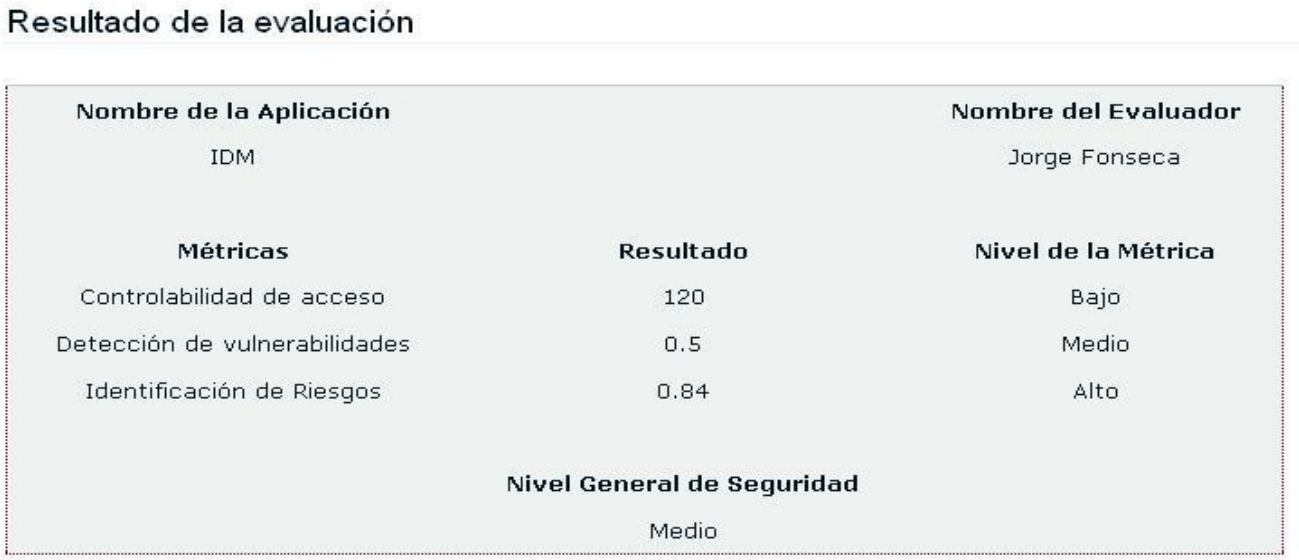
Descripción: Cuando los datos han sido transformados, se procede a cargar la información nuevamente a la BD. El sistema envía los datos a la tabla correspondiente en la BD.	
Validaciones	
Prototipo de interfaz de usuario	
Requisitos de calidad del servicio	✓ SGBD PostgreSQL 8.3.5 o superior ✓ Máquina virtual de Java ✓ Pentaho Data Integration ✓ Servidor web Apache

Tabla 23: Descripción de la tarea "Cargar datos del fichero"

- ✓ **Especificación de tareas del escenario: "Generar reporte".**

El escenario se dividió en las tareas: Mostrar reporte de evaluación y Exportar reporte a formato pdf.

Nombre de la tarea: Mostrar reporte de evaluación		**Identificador:** SC 5.1
Objetivo de la tarea: Mostrar al usuario los resultados del proceso de evaluación.		
Persona: Evaluador		
Iteración: 2da	**Prioridad:** Secundario	**Complejidad:** 1
Precondiciones: ✓ Debe existir un proyecto para evaluar, debe haber sido cargado el fichero con el resultado de las pruebas de seguridad.		
Descripción: El evaluador selecciona la opción resultado de la evaluación, el sistema lo re-direcciona a la página que contiene el reporte de la evaluación, la cual contiene una tabla con la información generada después de realizar el proceso de evaluación.		
Validaciones		
Prototipo de interfaz de usuario		



Requisitos de calidad del servicio	-

Tabla 24: Descripción de la tarea "Mostrar reporte de evaluación"

Nombre de la tarea: Exportar reporte a formato pdf		**Identificador:** SC 5.2
Objetivo de la tarea: Permitir al evaluador una mejor visualización del proceso de evaluación.		
Persona: Evaluador		
Iteración:1ra	**Prioridad:** Opcional	**Complejidad:** 1
Precondiciones ✓ Deben existir los resultados del proceso de evaluación.		
Descripción: El escenario comienza cuando el evaluador selecciona la opción exportar a pdf, el sistema guarda la información en el formato seleccionado.		
Validaciones		
Prototipo de interfaz de usuario		

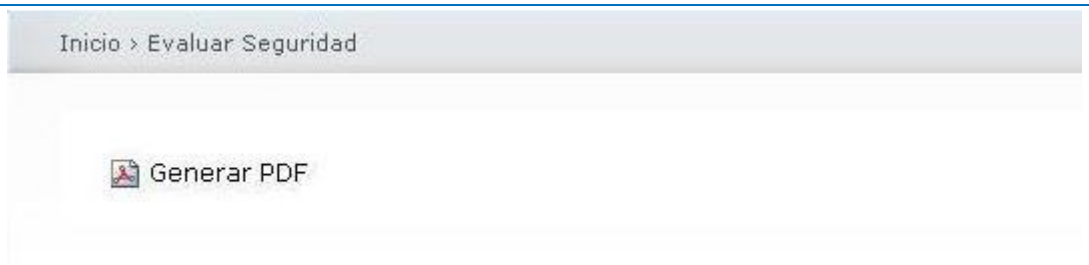
	
Requisitos de calidad del servicio	-

Tabla 25: Descripción de la tarea "Exportar reporte a formato pdf"

Anexo III: Patrones de Diseño

- ✓ **Bridge (Puente):** La capa de abstracción de bases de datos de Drupal se aplica de una forma similar al patrón de diseño Bridge. Los módulos necesitan ser escritos en una forma que es independiente del sistema que se está utilizando en la BD, y proporciona la capa de abstracción para ello. La nueva capa de BD se puede escribir que conforme la API definida por el puente, añadiendo soporte para más sistemas de BD sin la necesidad de modificar el código del módulo.

- ✓ **Chain of Responsibility (cadena de responsabilidades)**: El sistema de menús de Drupal sigue el patrón Chain of Reponsibility. En cada solicitud de la página, el menú del sistema determina si hay un módulo para gestionar la solicitud, si el usuario tiene acceso a los recursos solicitados, y que la función se llama para hacer el trabajo. Para ello, el mensaje se pasa a la opción del menú correspondiente a la vía de la solicitud. Si el elemento de menú no puede manejar la petición, se pasa de la cadena. Esto continúa hasta que un módulo se encarga de la petición, un módulo niega el acceso para el usuario, o la cadena se ha agotado.

Anexo IV: Descripción de las interfaces

- ✓ Interfaz "Autenticar usuario": permite que el evaluador se autentique para poder acceder al sistema

Figura 9: Interfaz "Autenticar usuario"

- ✓ Interfaz "Gestionar Proyectos": permite administrar todas las acciones que se realizan sobre los proyectos existentes en el sistema, así como la inserción de proyectos nuevos. Cuenta con las opciones Adicionar, Editar y Eliminar proyectos, además de listar los proyectos que existen en esos momentos y el estado actual de los mismos.

Figura 10: Interfaz "Gestionar Proyectos"

- ✓ Interfaz "Enviar Fichero de Prueba": permite enviar a la BD el fichero que contiene el resultado de las pruebas de seguridad para realizarle el proceso ETL.

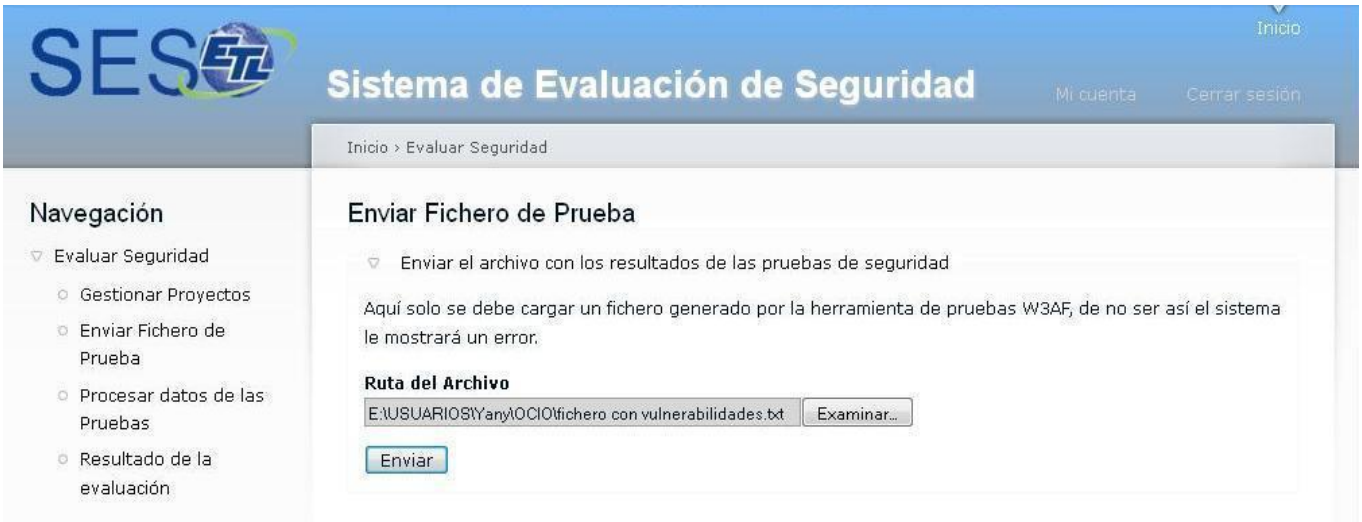


Figura 11: Interfaz "Enviar Fichero de Prueba"

- ✓ Interfaz “Procesar datos de las Pruebas”: es aquí donde el evaluador iniciará el proceso ETL que será aplicado al fichero almacenado en la BD.

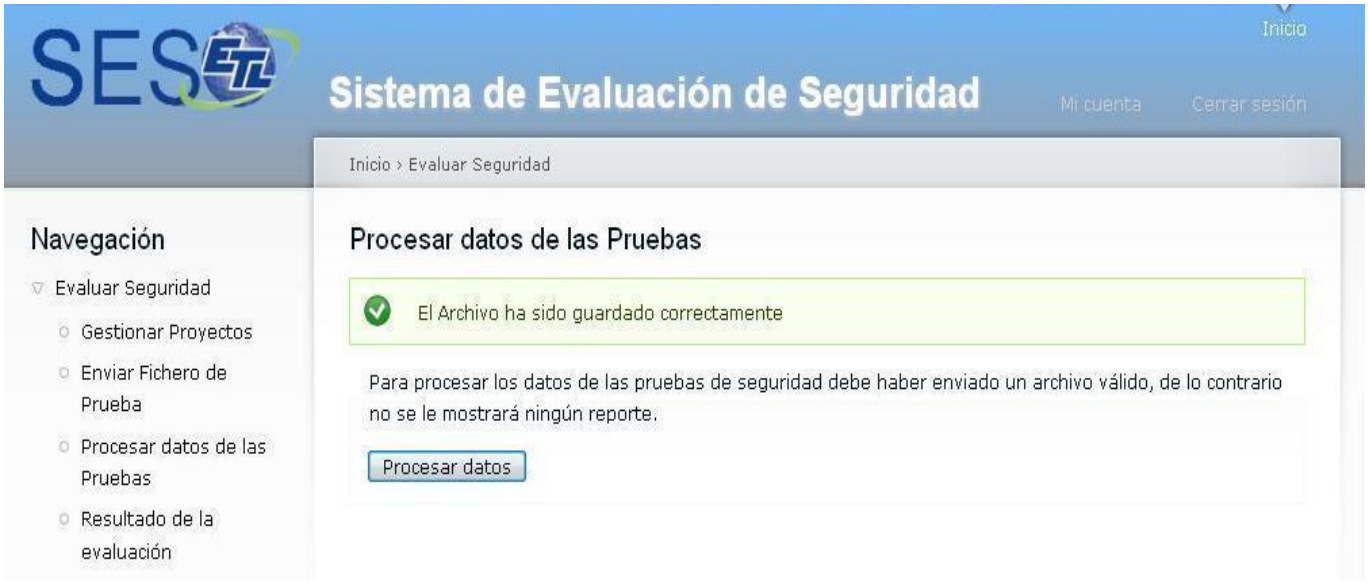


Figura 12: Interfaz “Procesar datos de las pruebas”

Anexo V: Pruebas unitarias

```
3   require_once ('PHPUnit/Framework.php');
4
5   class evaluacion.module.Test extends PHPUnit_Framework_TestCase {
6
7       public function test_calcular_vulnerabilidades()
8       {
9           $modulo_evaluacion = new Modulo_Evaluacion();
10          $this->assertEquals(0.8,$modulo_evaluacion->calcular_vulnerabilidades());
11      }
12  }
13      public function test_modulo_evaluacion_etl()
14      {
15          $modulo_evaluacion = new Modulo_Evaluacion();
16          $valor_esperado = 'No hay proyectos activos';
17          $this->assertEquals($valor_esperado,$modulo_evaluacion->modulo_evaluacion_etl());
18      }
19  }
20
```

Figura 13: Prueba unitaria a las funciones “calcular_vulnerabilidades" y "modulo_evaluacion”

Prueba de unidad		
Nombre de la prueba: *test*_calcular_vulnerabilidades		
Estado: Satisfactoria	**Tipo:** Caja Blanca	**Ultima ejecución:** 06/04/2012
Ejecutado por: Jorge Fonseca	**Verificado por:** Yayneris Zambrana	
Descripción: Para ejecutar la prueba es necesario haber insertado un proyecto ya que existen valores que son de utilidad en esta función. La misma se utiliza para determinar el nivel que presenta el sistema para la métrica "Detección de vulnerabilidades".		
Entrada: -		
Criterio de Aceptación: Se muestra en el resultado de la evaluación el nivel que presenta el sistema para esta métrica.		
Resultado 2 PHPUnit 3.6.10 by Sebastian Bergmann 3 4 . 5 6 Time 0 seconds, Memory: 5.50Mb 7 8 OK (1 test, 1 assertion) 9		

Tabla 26: Descripción de la prueba unitaria a la función "calcular_vulnerabilidades"

Prueba de unidad		
Nombre de la prueba: *test*_modulo_evaluacion_etl		
Estado: Satisfactoria	**Tipo:** Caja Blanca	**Ultima ejecución:** 06/04/2012
Ejecutado por: Jorge Fonseca	**Verificado por:** Yayneris Zambrana	
Descripción: Para ejecutar la prueba es necesario tener en BD el resultado de las pruebas. Esta función es la encargada de realizar el proceso ETL a los datos, seleccionando solo los necesarios para		

la evaluación.	
Entrada: -	
Criterio de Aceptación: Será mostrada la evaluación final.	
Resultado	

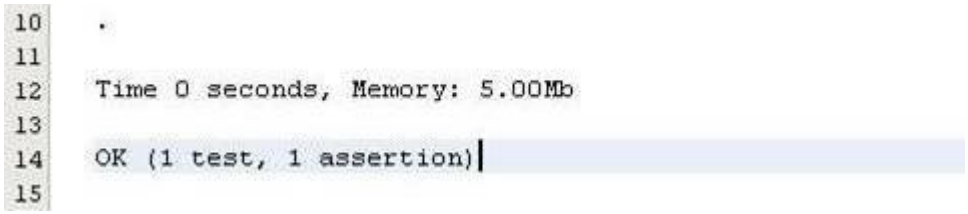


Tabla 27: Descripción de la prueba unitaria a la función "*test*_modulo_evaluacion_etl"

Anexo VI: Aplicación de Pruebas de Caja Negra

- ✓ **Escenario: "Autenticar usuario".**

Escenario	Descripción	Texto	Respuesta del Sistema	Flujo Central
Autenticar usuario	El evaluador debe autenticarse para poder acceder al sistema y realizar la evaluación.	Nombre de Usuario Contraseña (Válidos)	El sistema le permite al usuario acceder a las opciones del sistema.	Inicio de sección
		Nombre de Usuario Contraseña (Inválidos)	El sistema muestra un mensaje de error informando que existen datos incorrectos.	

Tabla 28: Descripción del caso de prueba al escenario "Autenticar usuario"

- ✓ **Escenario: "Cargar fichero".**

Tareas del Escenario	Descripción	Texto	Respuesta del Sistema	Flujo Central
Tarea: Seleccionar fichero	Se procede a seleccionar el fichero que posee el resultado de las pruebas de seguridad realizadas al *software* que será evaluado.	Ruta del archivo (Válidos)	En esta sección el sistema no muestra nada, solo carga la dirección del archivo seleccionado.	Opción Enviar Fichero de Prueba / Botón examinar.
		Ruta del archivo (Inválidos)	En este caso no sucede nada hasta que se pase a la otra sección de Subir el fichero.	
Tarea: Subir fichero	El fichero es cargado en la BD.	Ruta del archivo (Válidos)	El sistema inserta los datos del fichero en la BD	Opción Enviar Fichero de Prueba / Botón Enviar
		Ruta del archivo (Inválidos)	El sistema muestra un mensaje de error informando que los datos del fichero no fueron insertados en la BD correctamente	

			debido a que el archivo cargado no es el correcto.	

Tabla 29: Descripción del caso de prueba a las tareas del escenario "Cargar fichero"

- ✓ **Escenario: "Procesar datos del fichero".**

Tareas del Escenario	Descripción	Texto	Respuesta del Sistema	Flujo Central
Tarea: Extraer datos del fichero	Se procede extraer los datos del fichero que posee el resultado de las pruebas de seguridad.	Procesar datos (Válido: equivale a que se cargó un fichero a la BD)	El sistema inicia el proceso ETL.	Sección Procesar datos de las Pruebas**/** Botón Procesar datos
		Procesar datos (Inválido: equivale a que no se cargó un fichero a la BD)	El sistema informa que no existe fichero para procesar.	
Tarea: Transformar datos del fichero	Se procede a seleccionar los datos necesarios para la evaluación.		El sistema realizar esta sección una vez finalizada la anterior.	Se realiza de forma automática.
Tarea: Cargar datos del fichero	Son almacenados en la BD solo los resultados de la transformación.		El sistema realizar esta sección una vez finalizada la anterior.	Se realiza de forma automática.

Tabla 30: Descripción del caso de prueba a las tareas del escenario "Procesar datos del fichero"

- ✓ **Escenario: "Generar reporte".**

Tareas del Escenario	Descripción	Texto	Respuesta del Sistema	Flujo Central
Tarea: Mostrar reporte de evaluación	Se muestra al evaluador el reporte final donde se especifica el nivel de seguridad que posee su sistema.	El resultado de la evaluación es generado por el sistema automáticamente.	El sistema muestra los resultados arrojados después de realizar ETL y aplicar las métricas de seguridad a estos resultados.	Opción Reporte de evaluación.
			En caso de existir problema con los datos necesarios para la evaluación el sistema lo informará mediante un mensaje de error.	
Tarea: Exportar reporte a formato pdf	Esta tarea permite al evaluador exportar el resultado de la evaluación a pdf para una mejor visualización.		El sistema devuelve la evaluación en el formato especificado.	Opción Reporte de evaluación.

Tabla 31: Descripción del caso de prueba a las tareas del escenario "Generar reporte"